DOCKET NO: 245430US

IN THE UNITED STATES PATENT & TRADEMARK OFFICE

IN RE APPLICATION OF	:
HEBER MACMAHON, ET AL.	: EXAMINER: LAMPRECHT, J.
SERIAL NO: 10/721,827	:
FILED: NOVEMBER 26, 2003	: GROUP ART UNIT: 3737
FOR: AUTOMATED METHOD AND SYSTEM FOR THE EVALUATION OF DISEASE AND REGISTRATION ACCURACY IN THE SUBTRACTION OF TEMPORALLY SEQUENTIAL MEDICAL IMAGES	:

DECLARATION UNDER 37 C.F.R. § 1.131

COMMISSIONER FOR PATENTS
ALEXANDRIA, VIRGINIA 22313

SIR:

I, Heber MacMahon declare as follows:

1. I am Professor of Radiology at the University of Chicago and have performed original research in the area of computer-aided detection of anatomical abnormalities since at least 1987 at the Rossmann Laboratories for Radiologic Image Research.

2. In accordance with my employment with the University of Chicago, all of my inventions relating to the computer-aided detection of anatomical abnormalities were subject to assignment to the University of Chicago during all time intervals discussed herein.

3. By at least March 2003, I and my co-inventor had formed in our minds a definite and permanent idea of a complete and operative method and apparatus for determining whether a region of interest in a subtraction image includes an abnormality associated with

pathological change as currently appearing in at least Claims 1 and 22 of the above-identified application, the method including the steps of (a) obtaining a temporal subtraction image of an anatomical region of a patient from two images taken at respective times separated by a time interval that is long enough to allow for pathological change in the anatomical region; (b) extracting at least one feature from the subtraction image; and (c) determining whether a region of interest in the subtraction image includes an abnormality associated with the pathological change, based on the extracted at least one feature, wherein the determining step comprises distinguishing a region of pathologic change from regions with a misregistration artifact, as it would be built and applied in practice.

4. By at least March 2003, I and my co-inventor had formed in our minds a definite and permanent idea of a complete and operative method and apparatus for generating a temporal subtraction image as currently appearing in at least Claims 14 and 33 of the above-identified application, the method including the steps of (a) obtaining a first dual-energy image, a first standard image, and one of a first bone image and a first soft tissue image from the first dual-energy image at a first point in time; (b) obtaining a second dual-energy image, a second standard image, and one of a second bone image and a second soft tissue image from the second dual-energy image at a second point in time; (c) using the first and second standard images to obtain shift vectors to obtain image registration; and (d) performing temporal subtraction, using said shift vectors, on one of the first and second bone images or one of the first and second soft tissue images to produce a temporally subtracted image, as it would be built and applied in practice.

5. By at least March 2003, I and my co-inventor first reduced to practice and demonstrated the method and apparatus for determining whether a region of interest in a subtraction image includes an abnormality associated with pathological change as currently

appearing in at least Claims 1 and 22 of the above-identified application, as stated in the attached Invention Disclosure, which was submitted to the University of Chicago Office of Technology and Intellectual Property in August, 2003.

6. By at least March 2003, I and my co-inventor first reduced to practice and demonstrated the method and apparatus for generating a temporal subtraction image as currently appearing in at least Claims 14 and 33 of the above-identified application, as stated in the attached Invention Disclosure, which was submitted to the University of Chicago Office of Technology and Intellectual Property in August, 2003.

7. By at least March 11, 2003, I and my co-inventor first reduced to practice and demonstrated the method and apparatus for determining whether a region of interest in a subtraction image includes an abnormality associated with pathological change as currently appearing in at least Claims 1 and 22 of the above-identified application, as evidenced by the attached software program listing.

8. By at least March 11, 2003, I and my co-inventor first reduced to practice and demonstrated the method and apparatus for generating a temporal subtraction image as currently appearing in at least Claims 14 and 33 of the above-identified application, as evidenced by the attached software program listing.

9. On June 8, 2003, I gave a presentation entitled "Computer-Assisted Diagnosis: Breast and Thoracic Imaging," submitted herewith, at the 20th Symposium for Computer Applications in Radiology, Annual Meeting 2003 (Boston, MA), as shown by page 7 of the attached program of that conference. The presentation illustrates the method and apparatus as currently appearing in at least Claims 1, 14, 22, and 33 of the above-identified application.

10. I further declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that

these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the above-referenced application or any patent issuing thereon.

By: ______________________ Date: 3- 13- 09

Heber MacMahon
2144 N. Cleveland
Chicago, IL 60614

DOCKET NO: 245430US

IN THE UNITED STATES PATENT & TRADEMARK OFFICE

IN RE APPLICATION OF :

HEBER MACMAHON, ET AL. : EXAMINER: LAMPRECHT, J.

SERIAL NO: 10/721,827 :

FILED: NOVEMBER 26, 2003 : GROUP ART UNIT: 3737

FOR: AUTOMATED METHOD AND SYSTEM FOR THE EVALUATION OF DISEASE AND REGISTRATION ACCURACY IN THE SUBTRACTION OF TEMPORALLY SEQUENTIAL MEDICAL IMAGES :

DECLARATION UNDER 37 C.F.R. § 1.131

COMMISSIONER FOR PATENTS
ALEXANDRIA, VIRGINIA 22313

SIR:

I, Samuel G. Armato, III declare as follows:

1. I am an Associate Professor of Radiology and the Committee on Medical Physics at the University of Chicago and have performed original research in the area of computer-aided detection of anatomical abnormalities since at least 1991 at the Rossmann Laboratories for Radiologic Image Research.

2. In accordance with my employment with the University of Chicago, all of my inventions relating to the computer-aided detection of anatomical abnormalities were subject to assignment to the University of Chicago during all time intervals discussed herein.

3. By at least March 2003, I and my co-inventor had formed in our minds a definite and permanent idea of a complete and operative method and apparatus for determining

whether a region of interest in a subtraction image includes an abnormality associated with pathological change as currently appearing in at least Claims 1 and 22 of the above-identified application, the method including the steps of (a) obtaining a temporal subtraction image of an anatomical region of a patient from two images taken at respective times separated by a time interval that is long enough to allow for pathological change in the anatomical region; (b) extracting at least one feature from the subtraction image; and (c) determining whether a region of interest in the subtraction image includes an abnormality associated with the pathological change, based on the extracted at least one feature, wherein the determining step comprises distinguishing a region of pathologic change from regions with a misregistration artifact, as it would be built and applied in practice.

4. By at least March 2003, I and my co-inventor had formed in our minds a definite and permanent idea of a complete and operative method and apparatus for generating a temporal subtraction image as currently appearing in at least Claims 14 and 33 of the above-identified application, the method including the steps of (a) obtaining a first dual-energy image, a first standard image, and one of a first bone image and a first soft tissue image from the first dual-energy image at a first point in time; (b) obtaining a second dual-energy image, a second standard image, and one of a second bone image and a second soft tissue image from the second dual-energy image at a second point in time; (c) using the first and second standard images to obtain shift vectors to obtain image registration; and (d) performing temporal subtraction, using said shift vectors, on one of the first and second bone images or one of the first and second soft tissue images to produce a temporally subtracted image, as it would be built and applied in practice.

5. By at least March 2003, I and my co-inventor first reduced to practice and demonstrated the method and apparatus for determining whether a region of interest in a

subtraction image includes an abnormality associated with pathological change as currently appearing in at least Claims 1 and 22 of the above-identified application, as stated in the attached Invention Disclosure, which was submitted to the University of Chicago Office of Technology and Intellectual Property in August, 2003.

6. By at least March 2003, I and my co-inventor first reduced to practice and demonstrated the method and apparatus for generating a temporal subtraction image as currently appearing in at least Claims 14 and 33 of the above-identified application, as stated in the attached Invention Disclosure, which was submitted to the University of Chicago Office of Technology and Intellectual Property in August, 2003.

7. By at least March 11, 2003, I and my co-inventor first reduced to practice and demonstrated the method and apparatus for determining whether a region of interest in a subtraction image includes an abnormality associated with pathological change as currently appearing in at least Claims 1 and 22 of the above-identified application, as evidenced by the attached software program listing.

8. By at least March 11, 2003, I and my co-inventor first reduced to practice and demonstrated the method and apparatus for generating a temporal subtraction image as currently appearing in at least Claims 14 and 33 of the above-identified application, as evidenced by the attached software program listing.

9. On June 8, 2003, my co-inventor, Heber MacMahon, gave a presentation entitled "Computer-Assisted Diagnosis: Breast and Thoracic Imaging," submitted herewith, at the 20th Symposium for Computer Applications in Radiology, Annual Mccting 2003 (Boston, MA), as shown by page 7 of the attached program of that conference. The presentation illustrates the method and apparatus as currently appearing in at least Claims 1, 14, 22, and 33 of the above-identified application.

10. I further declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the above-referenced application or any patent issuing thereon.

By: ____________________ Date: 3/13/09

Samuel G. Armato, III, Ph.D.
8247 Cambridge Court
Downers Grove, IL 60516

Invention Disclosure

UCHI # 1144
8/27/03

University of Chicago Office of Technology and Intellectual Property
UCTech

Please Print or Type

1. TITLE OF INVENTION/SOFTWARE (What does it do? Exclude information about how it does it.)

AUTOMATED METHOD AND SYSTEM FOR THE VISUALIZATION OF PATHOLOGIC CHANGE DEMONSTRATED IN THE SUBTRACTION OF TEMPORALLY SEQUENTIAL MEDICAL IMAGES

2. INVENTOR(S)*Title

**Please attach curriculum vitae*

INVENTOR(S)*Title	Citizenship	Dept/Campus Address	Telephone #
Heber MacMahon, M.D.	Ireland	Department of Radiology	2-1604
Samuel G. Armato III, Ph.D.	USA	Department of Radiology	4-3044

3. CONTRIBUTORS TO THE INVENTION

Name/Title	Citizenship	Dept/Campus Address	Telephone #
Roger M. Engelmann, M.S.	USA	Department of Radiology	4-5093
Michalis Aristophanous, B.A.	Greece	Department of Radiology	4-5107
Charles L. Croteau, D.O.	USA	Department of Radiology	4-5107

4. DESCRIPTION OF INVENTION - Attach any additional information or background documentation. What does the invention do? How does it do it? What is the significance of the invention? How is it an improvement over the existing state-of-the-art?

The clinical utility of temporal subtraction images created from sequential chest radiographic images of the same patient has been demonstrated in the literature and through direct experience at The University of Chicago. A potential barrier to the wide-spread clinical use of this powerful technique is the unique appearance of temporal subtraction images, an appearance that some radiologists may find distracting. This invention provides a mechanism by which clinically significant information present in a temporal subtraction image may be intelligently integrated with the original radiographic image.

Are there any special time sensitivities for this invention? Yes____ No _X__ Please explain on separate sheet.

5. DISCLOSURE RECORD	DATE	REFERENCES & COMMENTS Use separate sheet if necessary. Attach copies
A. Record date you first conceived of the invention	8/02 – 3/03	various aspects of the invention were conceived throughout this period of time
B. First oral presentation of invention at seminars, meetings, conferences, etc.	N/A	
C. First publication, e.g. masters or doctoral theses, posters, articles, abstracts, seminars	N/A	
D. First successful demonstration, if any (reduction to practice)	3/03	
E. Other publications to date or anticipated publications	N/A	

6. SUPPORT

A. Inventions, discoveries or device-like software resulting from research or other activities carried out at the or with the substantial aid of its facilities or funds administered by it (including salary support) belong to the University. If this is not the case for the invention, please provide a memorandum substantiating a request that rights be assigned to the inventor(s). A memorandum is attached ____.

B. Was the invention/discovery conceived or first actually reduced to practice in the performance of work under any sponsored research or activity?
Yes _X__ No ____ Some aspects were, other aspects were not.

Sponsor	Contract/Grant #	Amount/Percent
NIH	CA64370-03	MacMahon (20%),Armato(10%)

C. Please designate whether any of the above is a training grant, funding awarded for educational purposes, or unrestricted gift funds.

D. Were any materials used in this invention obtained under a Materials Transfer Agreement?
Yes_____ No__X__ *If yes please indicate institution from which the materials were received and U of C PI involved in the agreement.*

7. BACKGROUND RESEARCH AND PRIOR ART related to your invention: (See attached instructions.)
___ No relevant prior art exists
___ See attached description
_X_ See attached publications or references

8. RECORDS to substantiate your invention's history includes: (See attached instructions)
_X__ Laboratory notes and records
___ Witnessed and dated?
___ Financial documents
___ Dated photographs
___ Idea only
_X_ Rough sketches/diagrams
___ Finished, working drawings
___ Other

9. TESTING which has already been conducted includes:
___ None
_X_ Functional testing of prototype to determine if it operates as intended
___ Market testing of invention

10. POTENTIAL LICENSEES
Please list any companies which you think may be interested in you invention.

Company	Address	Contact Person	Telephone #
R2 Technology, Inc.	Los Altos, CA		

11. TECHNICAL/MARKETING CONTACTS (See attached instructions)

Company	Address	Contact Person	Telephone #

12. I/We certify that the above information is complete and accurate:

8/15/03
Signature Date

8/19/03
Co-Inventor Signature Date

Co-Inventor Signature Date

13. The above confidential information was disclosed to and understood by:

Maryellen Giger 8-18-03
Witness' Signature Date

Maryellen L. Giger, Ph.D.
Witness' printed or typed name

8-25-03
Department Head Date

Richard L. Baron, M.D.
Department Head printed or typed name

THE UNIVERSITY OF CHICAGO

References

Kinsey H, Vannelli BD, Fontana R S, et al.: Application of digital image change detection to diagnosis and follow-up of cancer involving the lungs. Proc SPIE 70: 99-112, 1975.

Kano A, Doi K, MacMahon H, Hassell DD, Giger ML: Digital image subtraction of temporally subtracted chest images for detection of interval change. Med Phys 21: 453-461, 1994.

Sasaki Y, Katsuragawa S, Ishikawa I, MacMahon H, Doi K: Usefulness of temporally subtracted images in the detection of lung nodules in digital chest radiographs. Radiology 201: 400, 1996.

Difazio MC, MacMahon H, Xu XW, Tsai P, Shiraishi J, Armato SG Doi K: Digital chest radiography: Effect of temporal subtraction images on detection accuracy. Radiology 202: 447-452, 1997.

Sasaki Y, Katsuragawa S, MacMahon H, Doi K: Application of temporally subtracted images with mobile computed radiography system in screening chest radiographs. Proc CAR pp. 21-24, 1998

Katsuragawa S, Sasaki Y, MacMahon H, Ishida T, Doi K: Application of temporal subtraction to screening chest radiographs with a mobile computed radiography system. Proc 1st International Workshop on Computed-Aided Diagnosis, pp. 51-56, 1998.

Ishida T, Ashizawa K, Engelmann R, Katsuragawa S, MacMahon H, Doi K: Application of temporal subtraction for detection of interval changes in chest radiographs: Improvement of subtraction image using automated initial image matching. Journal of Digital Imaging 12: 77-86, 1999.

Ishida T, Katsuragawa S, Nakamura K, MacMahon H, Doi K: Iterative image warping technique for temporal subtraction of sequential chest radiographs to detect interval change. Med Phys 26: 1320-1329, 1999.

Katsuragawa S, Tagashira H, Li Q, MacMahon H, Doi K: Comparison of the quality of temporal subtraction images obtained with manual and automated methods of digital chest radiography. Journal of Digital Imaging 12: 166-172, 1999.

Nakata H, Nakamura T, Uozumi H, Watanabe T, Aoki, et al.: Clinical usefulness of temporal subtraction on digital chest radiographs. CARS 2000 pp. 793-797, 2000.

Uozumi T, Nakamura K, Watanabe H, Nakata H, Katsuragawa S, Doi K: ROC analysis of detection of metastatic pulmonary nodules on digital chest radiographs with temporal subtraction. Academic Radiology 8: 871-878, 2001.

Tsubamoto M, Johkoh T, Kozuka T, Tomiyama N, Hamada S, Honda O, Mihara N, Koyama M, Maeda M, Nakamura H, Fujiwara K: Temporal subtraction for the detection of hazy pulmonary opacities on chest radiography. AJR 179: 467-471, 2002.

Kakeda S, Nakamura K, Kamada K, Watanabe H, Nakata H, Katsuragawa S, Doi K: Improved detection of lung nodules by using a temporal subtraction technique. Radiology 224: 145-151, 2002.

Johkoh T, Kozuka T, Tomiyama N, Hamada S, Honda O, Mihara N, Koyama M, Tsubamoto M, Maeda M, Nakamura H, Saki H, Fujiwara K: Temporal subtraction for detection of solitary pulmonary nodules on chest radiographs: evaluation of a commercially available computer-aided diagnosis system. Radiology 223: 806-811, 2002.

CONFIDENTIAL INVENTION DISCLOSURE DOCUMENT

Instructions:

Completing this confidential document is the first step toward protecting and commercializing your invention.

The following instructions refer to the correspondingly numbered sections on the attached document.

1 Use a separate form for each invention. Title should convey what an invention is and what it does, not how it works.

2 Name as co-inventors only those who contributed significantly in the inventive concepts of the invention.

3 Name as contributors others who assisted with refining or reducing to practice some aspect of the invention.

4 Use simple language to describe your invention and define technical terms. State in a sentence or two what it does (which is public information), how it does it, and how it is an improvement over the existing state-of-the-art.

5 If you have published or presented this invention previously, please submit copies of the relevant abstracts and papers.

(B) and (C): The term "publication" as used here means written or oral communication, without restriction of confidentiality, which would enable members of the general public to legally gain access to a description of the invention. Publication therefore includes masters or doctoral theses, posters, articles, abstracts, and public seminars.

Publication to third parties more than one year before filing a U.S. patent application will preclude obtaining U.S. patent protection. Publication to third parties prior to the filing of a patent application will preclude foreign patenting.

(D) Reduction to practice occurs when an invention has been embodied in some physical form that is used to demonstrate its workability.

6 (A) "Where research or other activities carried out at the University or with the substantial aid of its facilities or funds administered by it result in inventions, discoveries, or device-like software, such products shall be disclosed to the University, shall be the property of the University and shall be assigned to the University or an organization designated by the University." If there is any doubt a disclosure should be filed with a memorandum indicating how the invention does not meet these conditions and requesting that the university provide the inventors with a formal release of rights.

(B) through (D) If the invention was "conceived or first actually reduced to practice in the performance of work under any funding agreement or other sponsored research," give the applicable contract or grant number(s). Indicate whether the grant was a training or other educational grant. If materials used in this invention were obtained under a Materials Transfer Agreement, provide the name of the institution or company from which materials were obtained.

7 Please provide a general description of the background research and pertinent prior work done in the field related to your invention. If possible, list five of the most relevant publications and attach copies if available.

8 Please note all documents or forms in which your invention has been recorded or described to substantiate your invention's history, such as lab notes. Whenever possible, it is desirable to have a witness sign such documents. Actual documents may be requested if patent application is filed.

9 Self explanatory.

10 Self explanatory.

11 Please list names and telephone numbers of individuals who possess technical or marketing knowledge related to your potential application that could assist in the commercial evaluation of the invention.

12 All inventors must sign and date the disclosure form. Only persons other than co-inventors who understand the invention may serve as witnesses; departmental colleagues are excellent resources.

Note: It is very helpful for UCTech and the inventors to complete a Revenue Distribution Agreement at the time of disclosure and submit it along with the disclosure. This can prevent problems in distributing income when an invention has been successfully licensed.

Also, if the disclosure precedes an imminent publication, you may fax or deliver it to UCTech without the signature of witnesses or the signature of the Department Head, as long as you follow up quickly with the properly signed documents.

PLEASE SUBMIT ALL COPIES TO:

Compliance Coordinator
UCTech
The University of Chicago Office of Technology and Intellectual Property
5640 South Ellis, Suite 405
Chicago, IL 60637

Re: Invention Disclosure

Phone: 773-702-1692
Fax: 773-702-0741

UCTech's mail code for interoffice mailings is **AAC405**

If you made a verbal disclosure to a specific Project Manager at UCTech, you can also send a copy of the disclosure to that Manager's attention.

Name	Type	Modified	Size	Ratio	Packed	Path
ApplySubParams.ii	II File	3/11/2003 2:30 PM	221	22%	173	src-ApplyTSParams\ii_files\
ApplyTSParams.C	C File	3/11/2003 2:30 PM	7,778	72%	2,207	src-ApplyTSParams\
ApplyTSParams.C~	C~ File	3/11/2003 2:30 PM	7,777	72%	2,205	src-ApplyTSParams\
ApplyTSParams.ii	II File	3/11/2003 2:30 PM	219	21%	172	src-ApplyTSParams\ii_files\
ApplyTSParams.o	O File	3/11/2003 2:30 PM	746,244	83%	128,714	src-ApplyTSParams\
CardiacEdgeScaling.C	C File	3/11/2003 2:30 PM	1,587	66%	543	TS-mainprogram\src\
CardiacEdgeScaling.o	O File	3/11/2003 2:30 PM	74,260	73%	19,698	TS-mainprogram\src\
CardiacParmScaling.C	C File	3/11/2003 2:30 PM	3,293	75%	814	TS-mainprogram\src\
CardiacParmScaling.o	O File	3/11/2003 2:30 PM	75,028	73%	20,088	TS-mainprogram\src\
CHEST_PACK3.f	F File	3/11/2003 2:30 PM	36,330	76%	8,618	TS-mainprogram\src\
CHEST_PACK3.o	O File	3/11/2003 2:30 PM	46,884	67%	15,238	TS-mainprogram\src\
chs_sub.f	F File	3/11/2003 2:30 PM	48,268	84%	7,783	TS-mainprogram\src\
chs_sub.o	O File	3/11/2003 2:30 PM	58,264	74%	15,004	TS-mainprogram\src\
clab.f	F File	3/11/2003 2:30 PM	7,598	78%	1,663	TS-mainprogram\src\
ContraSub.H	H File	3/11/2003 2:30 PM	75	31%	52	TS-mainprogram\src\
copy_image.C	C File	3/11/2003 2:30 PM	418	56%	184	TS-mainprogram\src\extra\
copy_sub.f	F File	3/11/2003 2:30 PM	476	66%	163	TS-mainprogram\src\
copy_sub.o	O File	3/11/2003 2:30 PM	4,800	65%	1,677	TS-mainprogram\src\
corr1.f	F File	3/11/2003 2:30 PM	4,956	70%	1,501	TS-mainprogram\src\
corr1.o	O File	3/11/2003 2:30 PM	8,484	62%	3,241	TS-mainprogram\src\
corrr.f	F File	3/11/2003 2:30 PM	7,287	72%	2,069	TS-mainprogram\src\
corrr1.f	F File	3/11/2003 2:30 PM	4,984	70%	1,504	TS-mainprogram\src\
corrr1.o	O File	3/11/2003 2:30 PM	8,492	62%	3,247	TS-mainprogram\src\
cqlt1.f	F File	3/11/2003 2:30 PM	2,166	68%	688	TS-mainprogram\src\
CTS_FAULT1.f	F File	3/11/2003 2:30 PM	10,737	76%	2,568	TS-mainprogram\src\
CTS_QUICK_Y3.f	F File	3/11/2003 2:30 PM	43,231	79%	9,270	TS-mainprogram\src\
CTS_QUICK_Y3.f~	F~ File	3/11/2003 2:30 PM	43,160	79%	9,260	TS-mainprogram\src\
CTS_QUICK_Y3.o	O File	3/11/2003 2:30 PM	26,728	67%	8,860	TS-mainprogram\src\
CTS_QUICK_Y3_VIEW.f	F File	3/11/2003 2:30 PM	40,621	81%	7,564	TS-mainprogram\src\extra\
CurveScaling.C	C File	3/11/2003 2:30 PM	1,413	63%	527	TS-mainprogram\src\
CurveScaling.o	O File	3/11/2003 2:30 PM	74,676	73%	19,893	TS-mainprogram\src\
DENSITY_CORRECTION.f	F File	3/11/2003 2:30 PM	1,822	71%	521	TS-mainprogram\src\
DENSITY_CORRECTION.o	O File	3/11/2003 2:30 PM	4,680	64%	1,668	TS-mainprogram\src\
DETERMINE_MIDLINE.f	F File	3/11/2003 2:30 PM	9,155	76%	2,176	TS-mainprogram\src\
DETERMINE_MIDLINE_C.f	F File	3/11/2003 2:30 PM	9,970	76%	2,371	TS-mainprogram\src\
DETERMINE_MIDLINE_C.o	O File	3/11/2003 2:30 PM	12,788	63%	4,757	TS-mainprogram\src\
diaphragm_detection.f	F File	3/11/2003 2:30 PM	31,549	78%	6,974	TS-mainprogram\src\
diaphragm_detection.o	O File	3/11/2003 2:30 PM	45,552	70%	13,648	TS-mainprogram\src\

Name	Type	Modified	Size	Ratio	Packed	Path
DicomUtil.c	C File	3/11/2003 2:30 PM	29,198	85%	4,315	TS-mainprogram\src\
DicomUtil.c~	C~ File	3/11/2003 2:30 PM	28,957	86%	4,191	TS-mainprogram\src\
DicomUtil.o	O File	3/11/2003 2:30 PM	52,136	69%	16,242	TS-mainprogram\src\
disp_image.C	C File	3/11/2003 2:30 PM	8,668	72%	2,425	TS-mainprogram\src\extra\
dsp_final.C	C File	3/11/2003 2:30 PM	2,041	74%	521	TS-mainprogram\src\extra\
dsp_fitting.C	C File	3/11/2003 2:30 PM	4,164	72%	1,171	TS-mainprogram\src\extra\
dsp_local_matching.C	C File	3/11/2003 2:30 PM	3,942	71%	1,142	TS-mainprogram\src\extra\
dsp_one_image.C	C File	3/11/2003 2:30 PM	1,217	68%	393	TS-mainprogram\src\extra\
dsp_ribcage_heart.C	C File	3/11/2003 2:30 PM	6,592	78%	1,470	TS-mainprogram\src\extra\
dsp_ROI.C	C File	3/11/2003 2:30 PM	4,827	72%	1,339	TS-mainprogram\src\extra\
dsp_shift_ribcage.C	C File	3/11/2003 2:30 PM	5,191	77%	1,201	TS-mainprogram\src\extra\
dsp_two_images.C	C File	3/11/2003 2:30 PM	3,764	76%	895	TS-mainprogram\src\extra\
dsp_wprev_sub.C	C File	3/11/2003 2:30 PM	2,005	67%	661	TS-mainprogram\src\extra\
epct2.f	F File	3/11/2003 2:30 PM	1,984	66%	672	TS-mainprogram\src\
epct2.o	O File	3/11/2003 2:30 PM	6,120	65%	2,126	TS-mainprogram\src\
ersr11.f	F File	3/11/2003 2:30 PM	1,992	70%	600	TS-mainprogram\src\
exnb2.f	F File	3/11/2003 2:30 PM	1,873	67%	609	TS-mainprogram\src\
exnb2.o	O File	3/11/2003 2:30 PM	5,784	69%	1,799	TS-mainprogram\src\
fcrsubUtil.c	C File	3/11/2003 2:30 PM	843	60%	337	TS-mainprogram\src\
fcrsubUtil.o	O File	3/11/2003 2:30 PM	24,512	69%	7,616	TS-mainprogram\src\
fdpair.c	C File	3/11/2003 2:30 PM	12,106	77%	2,785	TS-mainprogram\src\
fname_printQ.c	C File	3/11/2003 2:30 PM	2,338	73%	623	TS-mainprogram\src\
fname_printQ.o	O File	3/11/2003 2:30 PM	23,296	69%	7,322	TS-mainprogram\src\
GET_DENSITY_CORRECTION_FACTOR.f	F File	3/11/2003 2:30 PM	14,651	76%	3,589	TS-mainprogram\src\
GET_DENSITY_CORRECTION_FACTOR.o	O File	3/11/2003 2:30 PM	19,408	65%	6,717	TS-mainprogram\src\
get_file_name.c	C File	3/11/2003 2:30 PM	2,334	68%	736	TS-mainprogram\src\
GET_PARAMETERS_SKIP.f	F File	3/11/2003 2:30 PM	7,037	72%	1,977	TS-mainprogram\src\
GET_PARAMETERS_SKIP.o	O File	3/11/2003 2:30 PM	6,760	63%	2,512	TS-mainprogram\src\
image.h	H File	3/11/2003 2:30 PM	14,108	68%	4,456	TS-mainprogram\src\
image.h~	H~ File	3/11/2003 2:30 PM	14,070	68%	4,450	TS-mainprogram\src\
ImageScaling.C	C File	3/11/2003 2:30 PM	6,405	87%	863	TS-mainprogram\src\
ImageScaling.o	O File	3/11/2003 2:30 PM	25,456	70%	7,659	TS-mainprogram\src\
init_match.f	F File	3/11/2003 2:30 PM	34,443	82%	6,037	TS-mainprogram\src\
init_match.o	O File	3/11/2003 2:30 PM	44,700	74%	11,426	TS-mainprogram\src\
init_match_rescue.f	F File	3/11/2003 2:30 PM	22,321	84%	3,582	TS-mainprogram\src\
INITIALIZATION.f	F File	3/11/2003 2:30 PM	1,504	75%	372	TS-mainprogram\src\
INITIALIZATION.o	O File	3/11/2003 2:30 PM	4,560	65%	1,613	TS-mainprogram\src\
INVERSED.f	F File	3/11/2003 2:30 PM	3,307	74%	876	TS-mainprogram\src\

Name	Type	Modified	Size	Ratio	Packed	Path
INVERSED.o	O File	3/11/2003 2:30 PM	7,316	65%	2,585	TS-mainprogram\src\
KOFITC_PACK.f	F File	3/11/2003 2:30 PM	4,639	63%	1,728	TS-mainprogram\src\
KOFITC_PACK.o	O File	3/11/2003 2:30 PM	14,900	64%	5,352	TS-mainprogram\src\
kofitc2.f	F File	3/11/2003 2:30 PM	3,209	54%	1,475	TS-mainprogram\src\
kofitc2.o	O File	3/11/2003 2:30 PM	13,052	65%	4,565	TS-mainprogram\src\
kyoukai.f	F File	3/11/2003 2:30 PM	1,253	62%	478	TS-mainprogram\src\
kyoukai.o	O File	3/11/2003 2:30 PM	5,460	63%	1,996	TS-mainprogram\src\
LATERAL_INCLINATION.f	F File	3/11/2003 2:30 PM	6,652	75%	1,654	TS-mainprogram\src\
line.f	F File	3/11/2003 2:30 PM	13,608	91%	1,276	TS-mainprogram\src\
line3.f	F File	3/11/2003 2:30 PM	14,359	91%	1,340	TS-mainprogram\src\
line3.o	O File	3/11/2003 2:30 PM	6,684	64%	2,422	TS-mainprogram\src\
list_file.c	C File	3/11/2003 2:30 PM	6,863	75%	1,716	TS-mainprogram\src\
LOCAL_MATCHING_SKIP.f	F File	3/11/2003 2:30 PM	10,610	75%	2,653	TS-mainprogram\src\
LOCAL_MATCHING_SKIP.o	O File	3/11/2003 2:30 PM	11,092	62%	4,202	TS-mainprogram\src\
lung_boundary.f	F File	3/11/2003 2:30 PM	5,168	76%	1,233	TS-mainprogram\src\
lung_boundary.o	O File	3/11/2003 2:30 PM	10,168	67%	3,368	TS-mainprogram\src\
LutFileName.c	C File	3/11/2003 2:30 PM	664	58%	276	TS-mainprogram\src\
LutFileName.o	O File	3/11/2003 2:30 PM	21,188	69%	6,495	TS-mainprogram\src\
Makefile	File	3/11/2003 2:30 PM	803	44%	451	src-ApplyTSParams\
Makefile	File	3/11/2003 2:30 PM	738	46%	400	src-ScaleUpPreprocShift\
Makefile	File	3/11/2003 2:30 PM	6,365	74%	1,668	TS-mainprogram\src\
Makefile	File	3/11/2003 2:30 PM	7,822	77%	1,819	TS-mainprogram\src\extra\
Makefile~	File	3/11/2003 2:30 PM	788	44%	442	src-ApplyTSParams\
Makefile~	File	3/11/2003 2:30 PM	788	44%	442	src-ScaleUpPreprocShift\
Makefile~	File	3/11/2003 2:30 PM	6,295	74%	1,639	TS-mainprogram\src\
makeVersionInfo	File	3/11/2003 2:30 PM	41	0%	41	TS-mainprogram\src\
makeVersionInfo~	File	3/11/2003 2:30 PM	18	0%	18	TS-mainprogram\src\
ModFCRDcmPatName.C	C File	3/11/2003 2:30 PM	2,333	64%	850	TS-mainprogram\src\
ModImageDirection.C	C File	3/11/2003 2:30 PM	3,873	72%	1,103	TS-mainprogram\src\
newribcage.f	F File	3/11/2003 2:30 PM	40,743	88%	4,840	TS-mainprogram\src\
NonlinearDensityCorrection.c	C File	3/11/2003 2:30 PM	2,854	73%	759	TS-mainprogram\src\
NonlinearDensityCorrection.o	O File	3/11/2003 2:30 PM	26,396	68%	8,331	TS-mainprogram\src\
null_string.f	F File	3/11/2003 2:30 PM	1,671	66%	567	TS-mainprogram\src\
null_string.o	O File	3/11/2003 2:30 PM	7,024	65%	2,452	TS-mainprogram\src\
PutDiscOnImage.C	C File	3/11/2003 2:30 PM	792	63%	296	TS-mainprogram\src\extra\
PutLineOnImage.C	C File	3/11/2003 2:30 PM	1,395	64%	506	TS-mainprogram\src\extra\
PutRectOnImage.C	C File	3/11/2003 2:30 PM	810	60%	322	TS-mainprogram\src\extra\
quadrant_histogram.C	C File	3/11/2003 2:30 PM	1,305	63%	483	TS-mainprogram\src\extra\

Name	Type	Modified	Size	Ratio	Packed	Path
quanti_mess_up.f	F File	3/11/2003 2:30 PM	321	50%	160	src-ApplyTSParams\
quanti_modify.f	F File	3/11/2003 2:30 PM	454	60%	182	src-ApplyTSParams\
quanti_modify.f~	F~ File	3/11/2003 2:30 PM	320	51%	158	src-ApplyTSParams\
quanti_modify.o	O File	3/11/2003 2:30 PM	4,792	65%	1,699	src-ApplyTSParams\
quanti2.f	F File	3/11/2003 2:30 PM	24,062	82%	4,383	TS-mainprogram\src\
quanti2.o	O File	3/11/2003 2:30 PM	25,068	70%	7,522	TS-mainprogram\src\
read_images_skip.c	C File	3/11/2003 2:30 PM	18,113	81%	3,469	TS-mainprogram\src\
read_images_skip.c~	C~ File	3/11/2003 2:30 PM	17,982	81%	3,447	TS-mainprogram\src\
read_images_skip.o	O File	3/11/2003 2:30 PM	39,440	68%	12,737	TS-mainprogram\src\
READ_ORIGINAL_IMAGES_SKIP.f	F File	3/11/2003 2:30 PM	2,721	73%	724	TS-mainprogram\src\
READ_ORIGINAL_IMAGES_SKIP.o	O File	3/11/2003 2:30 PM	6,012	65%	2,130	TS-mainprogram\src\
readTable.c	C File	3/11/2003 2:30 PM	737	51%	363	TS-mainprogram\src\
readTable.o	O File	3/11/2003 2:30 PM	21,512	69%	6,685	TS-mainprogram\src\
rescue.c	C File	3/11/2003 2:30 PM	1,777	64%	632	TS-mainprogram\src\
ReverseValue.C	C File	3/11/2003 2:30 PM	1,038	60%	420	src-ApplyTSParams\
ReverseValue.C	C File	3/11/2003 2:30 PM	1,126	60%	449	TS-mainprogram\src\
ReverseValue.C~	C~ File	3/11/2003 2:30 PM	1,126	60%	449	src-ApplyTSParams\
ReverseValue.o	O File	3/11/2003 2:30 PM	15,304	67%	4,982	src-ApplyTSParams\
ReverseValue.o	O File	3/11/2003 2:30 PM	73,908	74%	19,541	TS-mainprogram\src\
RIBCAGE_DETECTION.f	F File	3/11/2003 2:30 PM	12,620	74%	3,230	TS-mainprogram\src\
RIBCAGE_DETECTION.o	O File	3/11/2003 2:30 PM	19,708	69%	6,044	TS-mainprogram\src\
RIBCAGEPOINT3.f	F File	3/11/2003 2:30 PM	40,112	78%	8,774	TS-mainprogram\src\
RIBCAGEPOINT3.o	O File	3/11/2003 2:30 PM	59,552	70%	17,725	TS-mainprogram\src\
RibFeatureScaling.C	C File	3/11/2003 2:30 PM	1,082	60%	437	TS-mainprogram\src\
RibFeatureScaling.o	O File	3/11/2003 2:30 PM	74,008	73%	19,634	TS-mainprogram\src\
ROI_SELECTION.f	F File	3/11/2003 2:30 PM	17,041	77%	3,895	TS-mainprogram\src\
ROI_SELECTION.o	O File	3/11/2003 2:30 PM	17,112	64%	6,235	TS-mainprogram\src\
RotAngleByRibcage.C	C File	3/11/2003 2:30 PM	4,307	69%	1,349	TS-mainprogram\src\
RotAngleByRibcage.o	O File	3/11/2003 2:30 PM	76,560	73%	20,740	TS-mainprogram\src\
rotate.C	C File	3/11/2003 2:30 PM	3,155	66%	1,059	src-ApplyTSParams\
rotate.C	C File	3/11/2003 2:30 PM	3,270	66%	1,107	TS-mainprogram\src\
rotate.C~	C~ File	3/11/2003 2:30 PM	3,270	66%	1,107	src-ApplyTSParams\
rotate.o	O File	3/11/2003 2:30 PM	22,548	67%	7,436	src-ApplyTSParams\
rotate.o	O File	3/11/2003 2:30 PM	22,444	67%	7,309	TS-mainprogram\src\
ROTATE_IMAGE.f	F File	3/11/2003 2:30 PM	5,802	69%	1,826	TS-mainprogram\src\
runsub.pl	PL File	3/11/2003 2:30 PM	43,109	77%	10,114	scripts\
SAVE_DATAQ.f	F File	3/11/2003 2:30 PM	6,058	73%	1,615	TS-mainprogram\src\
SAVE_DATAQ.o	O File	3/11/2003 2:30 PM	11,784	66%	4,035	TS-mainprogram\src\

Name	Type	Modified	Size	Ratio	Packed	Path
save_image.C	C File	3/11/2003 2:30 PM	689	56%	301	TS-mainprogram\src\extra\
SAVE_SUBTRACTION_IMAGEQ.f	F File	3/11/2003 2:30 PM	1,651	75%	414	TS-mainprogram\src\
SAVE_SUBTRACTION_IMAGEQ.o	O File	3/11/2003 2:30 PM	4,904	64%	1,748	TS-mainprogram\src\
ScaleUpPreprocShift.C	C File	3/11/2003 2:30 PM	6,271	74%	1,649	src-ScaleUpPreprocShift\
ScaleUpPreprocShift.C~	C~ File	3/11/2003 2:30 PM	6,270	74%	1,649	src-ScaleUpPreprocShift\
ScaleUpPreprocShift.o	O File	3/11/2003 2:30 PM	223,588	75%	56,845	src-ScaleUpPreprocShift\
shift.C	C File	3/11/2003 2:30 PM	1,273	61%	491	src-ApplyTSParams\
shift.C	C File	3/11/2003 2:30 PM	1,384	61%	537	TS-mainprogram\src\
shift.C~	C~ File	3/11/2003 2:30 PM	1,273	62%	490	src-ApplyTSParams\
shift.C~	C~ File	3/11/2003 2:30 PM	1,384	61%	536	TS-mainprogram\src\
shift.o	O File	3/11/2003 2:30 PM	21,680	68%	6,913	src-ApplyTSParams\
shift.o	O File	3/11/2003 2:30 PM	21,652	68%	6,918	TS-mainprogram\src\
SHIFT_MAP_FITTING_INTP_Y3.f	F File	3/11/2003 2:30 PM	35,779	82%	6,615	TS-mainprogram\src\
SHIFT_MAP_FITTING_INTP_Y3.o	O File	3/11/2003 2:30 PM	29,416	68%	9,273	TS-mainprogram\src\
simple_subtraction.f	F File	3/11/2003 2:30 PM	7,026	77%	1,648	TS-mainprogram\src\
smooth.f	F File	3/11/2003 2:30 PM	649	50%	327	TS-mainprogram\src\
smooth.o	O File	3/11/2003 2:30 PM	4,692	64%	1,667	TS-mainprogram\src\
sobel2.f	F File	3/11/2003 2:30 PM	1,197	65%	424	TS-mainprogram\src\
sobel2.o	O File	3/11/2003 2:30 PM	6,244	64%	2,259	TS-mainprogram\src\
subroutine_file.txt	Text Document	3/11/2003 2:30 PM	4,250	68%	1,368	TS-mainprogram\src\
subroutine_name.txt	Text Document	3/11/2003 2:30 PM	4,100	66%	1,404	TS-mainprogram\src\
subroutine_tree.txt	Text Document	3/11/2003 2:30 PM	4,142	74%	1,095	TS-mainprogram\src\
TempSub.H	H File	3/11/2003 2:30 PM	7,209	73%	1,974	TS-mainprogram\src\
TempSubUtil.c	C File	3/11/2003 2:30 PM	46,133	87%	6,136	TS-mainprogram\src\
TempSubUtil.c.bak	BAK File	3/11/2003 2:30 PM	46,133	87%	6,136	TS-mainprogram\src\
TempSubUtil.o	O File	3/11/2003 2:30 PM	69,336	69%	21,313	TS-mainprogram\src\
TOP_AND_BOTTOM_LUNG.f	F File	3/11/2003 2:30 PM	3,998	68%	1,294	TS-mainprogram\src\
tsub_q.c	C File	3/11/2003 2:30 PM	3,229	71%	927	TS-mainprogram\src\
tsub_q.c~	C~ File	3/11/2003 2:30 PM	3,068	72%	873	TS-mainprogram\src\
tsub_q.o	O File	3/11/2003 2:30 PM	26,292	68%	8,506	TS-mainprogram\src\
WARP_AND_SUBTRACTION.f	F File	3/11/2003 2:30 PM	8,011	76%	1,904	src-ApplyTSParams\
WARP_AND_SUBTRACTION.f	F File	3/11/2003 2:30 PM	7,902	76%	1,861	TS-mainprogram\src\
WARP_AND_SUBTRACTION.f~	F~ File	3/11/2003 2:30 PM	7,902	76%	1,861	src-ApplyTSParams\
WARP_AND_SUBTRACTION.o	O File	3/11/2003 2:30 PM	10,256	63%	3,797	src-ApplyTSParams\
WARP_AND_SUBTRACTION.o	O File	3/11/2003 2:30 PM	9,536	63%	3,544	TS-mainprogram\src\
warp_rib_chs.f	F File	3/11/2003 2:30 PM	9,338	82%	1,667	TS-mainprogram\src\
warp_rib_chs.o	O File	3/11/2003 2:30 PM	10,724	67%	3,502	TS-mainprogram\src\
WEIGHTED_FIT_INTP.f	F File	3/11/2003 2:30 PM	13,872	79%	2,849	TS-mainprogram\src\

Name	Type	Modified	Size	Ratio	Packed	Path
WEIGHTED_FIT_INTP.o	O File	3/11/2003 2:30 PM	20,764	67%	6,840	TS-mainprogram\src\
WEIGHTED_NOFIT_INTP.f	F File	3/11/2003 2:30 PM	5,956	72%	1,668	TS-mainprogram\src\
WEIGHTED_NOFIT_INTP.o	O File	3/11/2003 2:30 PM	9,896	67%	3,281	TS-mainprogram\src\
write_headerless.c	C File	3/11/2003 2:30 PM	1,179	49%	603	TS-mainprogram\src\
write_headerless.c~	C~ File	3/11/2003 2:30 PM	636	40%	381	TS-mainprogram\src\
write_headerless.o	O File	3/11/2003 2:30 PM	21,016	68%	6,734	TS-mainprogram\src\
write_imageQ.c	C File	3/11/2003 2:30 PM	10,045	77%	2,295	TS-mainprogram\src\
write_imageQ.o	O File	3/11/2003 2:30 PM	32,348	67%	10,580	TS-mainprogram\src\
writeimagefloat.c	C File	3/11/2003 2:30 PM	369	36%	237	TS-mainprogram\src\
writeimagefloat.c~	C~ File	3/11/2003 2:30 PM	764	60%	306	TS-mainprogram\src\
writeimagefloat.o	O File	3/11/2003 2:30 PM	15,136	68%	4,886	TS-mainprogram\src\
writepreprocvals.c	C File	3/11/2003 2:30 PM	547	54%	251	TS-mainprogram\src\
writepreprocvals.c~	C~ File	3/11/2003 2:30 PM	369	36%	237	TS-mainprogram\src\
writepreprocvals.o	O File	3/11/2003 2:30 PM	15,252	68%	4,913	TS-mainprogram\src\
204 file(s)			**3,742,409**	**75%**	**926,596**	

```
c ***** vvvvvvv ********************************************************
c
c       Name : chest_pack3.for Verson 2.0
c
c       modified from verson 2.0(chest_pack2.for) by Xin-Wei Xu
c
c       Improving the checking and detection of top lung position
c
c       4/22/93
c
c ***** vvvvvvv ********************************************************

C**********************************************************************
C
C           Program of calculation of image profile
C
c           History: March 2,1992  original
C
C           Name: Profile_im_sub.FOR
C
C**********************************************************************
C
C
      SUBROUTINE profile_im_sub(image,profile,nxw,nyh,X1,X2,Y1,Y2,
     &                               CONTROL)
C
C
      IMPLICIT integer*2 (i-n)

      integer*2  nxw,nyh, !real image size, maxi size:1000 X 1215
     &  x1,x2,            !column range for profile calculation
     &  y1,y2,            !line range for profile calculation
     &  control           !=1,hori. profile; =2, vert. profile
      integer*2          image(nxw,nyh)  !image buffer

      REAL profile(nyh)  !nyh >= nxw

      real*8 sum

C*** start of program *****

      DO I=1,nyh
        profile(I)=0.0
      END DO

      sum=0.

      IF (CONTROL.EQ.1) THEN !horizontal profile of image
        DO IX=X1,X2
          DO IY=Y1,Y2
            sum=sum+FLOAT(image(IX,IY))
          END DO
          sum=sum/FLOAT(Y2-Y1+1)
          profile(IX)=sum
          sum=0.
        END DO
       ELSE                   !vertical profile of image
        DO IY=Y1,Y2
          DO IX=X1,X2
            sum=sum+FLOAT(image(IX,IY))
          END DO
          sum=sum/FLOAT(X2-X1+1)
          profile(IY)=sum
```

```
          sum=0.
        END DO
      END IF
C
C
      RETURN
      END
c*****************************************************
c
c    Program of smoothing profile with
c    N pixel average, N must be 3,5,7,9,11....
c
c    History: April 1,1992
c
c    Name: prof_smo_sub.for
c
c*****************************************************
c
c
        subroutine prof_smo_sub(prof,nw,nwr,N,prof_smo)
c
c
        implicit integer*2 (i-n)

        integer*2
     &      nw,            !dimension of input & output profiles
     &      nwr,           !real size of the profile content
     &      N              !number of PV for profile smooth,3,5,7,9,11,13,...

        real
     &    prof(nw),      !original input profile
     &    prof_smo(nw)   !smoothed output profile

        real*8 sum

c***** program begin ********

        m2=N-1
        m=int(float(m2)/2.0)  ! 2*m+1=N

        do j=m+1, nwr-m
          k1=j-m
          k2=j+m
          sum=0.0
          do k=k1,k2
            sum=sum+prof(k)
          end do
          prof_smo(j)=sum/float(N)
        end do

        do j=1,m
          prof_smo(j)=prof_smo(m+1)
        end do

        do j=nwr-m+1,nwr
          prof_smo(j)=prof_smo(nwr-m)
        end do

        return
        end
c**************************************************************
c
c    Program to obtain first derivative of profile
```

```
c
c    History: March 8, 1992
c             modified 4/10/93
c
c    Name: fd_south_sub.for
c
c**************************************************************
c
c
        subroutine fd_south_sub(prof,posi,fd,nw,nwr,
     &                increament,number_fd)

        implicit integer*2 (i-n)

        integer*2
     &    nw,           !dimension of input profile
     &    nwr,          !real size of the profile content, nwr<=nw
     &    increament,   !increament (in PV) for F.D. calculation
     &    number_fd,    !dimension of F.D. under the increament
     &    posi(nw)      !position of each F.D. under the increament

        real
     &    prof(nw),     !input original profile
     &    fd(nw)        !first derivative of prof. under the incre.

c ***** program begin ******

        inc2=2*increament

        k=0
        do i=1,nwr, increament
          k=k+1
          j1=i
          j2=j1+increament
          j3=j2+increament
          if (j3.gt.nwr) j3=nwr
          posi(k)=j2
          fd(k)=(prof(j3)-prof(j1))/float(inc2)  !$$$$$$$ South
        end do
        number_fd=k

!               fd is normalized

        return
        end

c**************************************************************
c
c    Program to obtain first derivative of profile
c
c    History: May 9, 1992
c             modified 4/10/93
c
c    Name: fd_north_sub.for
c
c**************************************************************
c
c
        subroutine fd_north_sub(prof,posi,fd,nw,nwr,
     &                increament,number_fd)

        implicit integer*2 (i-n)
```

```
        integer*2
     &    nw,           !dimension of input profile
     &    nwr,          !real size of the profile content, nwr<=nw
     &    increament,   !increament (in PV) for F.D. calculation
     &    number_fd,    !dimension of F.D. under the increament
     &    posi(nw)      !position of each F.D. under the increament

        real
     &    prof(nw),     !input original profile
     &    fd(nw)        !first derivative of prof. under the incre.

c ***** program begin ******

        inc2=2*increament

        k=0
        do i=1,nwr, increament
          k=k+1
          j1=i
          j2=j1+increament
          j3=j2+increament
          if (j3.gt.nwr) j3=nwr
          posi(k)=j2
          fd(k)=(prof(j1)-prof(j3))/float(inc2)      !$$$$$$ North
        end do
        number_fd=k

!               fd is normalized

        return
        end
c**************************************************************
c
c    Program to obtain second derivative of profile
c
c    History: March 8, 1992
c
c    Name: sd_south_sub.for
c
c*************************************************************
c
c
        subroutine sd_south_sub(prof,posi,sd,nw,nwr,
     &                increament,number_sd)

        implicit integer*2 (i-n)

        integer*2
     &    nw,           !dimension of input profile
     &    nwr,          !real size of profile content, nwr<=nw
     &    increament,   !increament (in PV) for S.D. calculation
     &    number_sd,    !dimension of S.D. under the increament
     &    posi(nw)      !position of each S.D. under the increament

        real
     &    prof(nw),     !input original profile
     &    sd(nw)        !second derivative of prof. under the incre.

c ***** program begin ******

        k=0
        do i=1,nwr, increament
          k=k+1
          j1=i
          j2=j1+increament
          if (j2.gt.nwr) then
            number_sd=k-1
            go to 100
          end if
          j3=j2+increament
          if (j3.gt.nwr) j3=nwr
          posi(k)=j2
          sd(k)=prof(j1)+prof(j3)-2.0*prof(j2)
          sd(k)=sd(k)
        end do
        number_sd=k

100     continue

        return
        end
c****************************************
c
c   Program for obtain a straight line
c   from two points (y=a+bx,or x=a+by)
c
c   Name: Straight_line_sub.for
c
c   History: March 5, 1992
c
c   summary: fit a line in two direction: Hori or Vert
c
c   The two points were put into program following the rule:
c   up then down (for Vert. cases),
c   and left then right (for Hori. cases).
c
c********************************************
c
c
        subroutine straight_line_sub(x1,y1,x2,y2,x,y,N,control)

        implicit integer*2 (i-n)

        integer*2
     &    x1,y1,       !position of first point (in PV)
     &    x2,y2,       !position of second point (in PV)
     &    N,           !dimension of the line
     &    x(N),        !x position of line(1-N, for Hori. line)
     &    y(N),        !y position of line(1-N, for Vert. line)
     &    control      !=1,horizontal line; =2, vertical line
                       !=1,hori:X=1->width; =2,vert:Y=1->height;

c**** program begin **********

        if (control.eq.1) then   ! horizontal straight line
          slope=float(y2-y1)/float(x2-x1)  !slope
          do i=1,N
            x(i)=i
            y(i)=y1+int(slope*float(i-x1))
          end do
        end if

        if (control.eq.2) then   ! vertical straight line
          slope=float(x1-x2)/float(y1-y2)  !slope
          do i=1,N
            y(i)=i
```

```
            x(i)=x1+int(slope*float(i-y1))
          end do
        end if

        return
        end
c****************************************
c
c   Program to obtain the crossing
c   point of two straight lines
c
c   Name: cross_p_2_line_sub.for
c
c   History: March 20, 1992
c
c
c********************************************
c
c
        subroutine cross_p_2_line_sub(x1,y1,N1,con1,x2,y2,N2,con2,x,y)

        implicit integer*2 (i-n)

        integer*2
     &    N1,                   ! dimension of line 1
     &    N2,                   ! dimension of line 2
     &    x1(N1),y1(N1),        ! line 1, x and y position
     &    x2(N2),y2(N2),        ! line 2, x and y position
     &    x, y,                 ! position of corssing point
     &    con1,           !=1,line 1 is hori; y1=a+b*x1; x1:1 -> width
                          !=2,line 1 is vert; x1=a+b*y1; y1:1 -> height

     &    con2            !=1,line 2 is hori; y2=a+b*x2; x2:1 -> width
                          !=2,line 2 is vert; x2=a+b*y2; y2:1 -> height

c**** program begin **********

        if ((con1.eq.1).and.(con2.eq.1)) then
          ix1=int(float(N1)/4.0)
          ix2=int(3.0*float(N1)/4.0)
          iy1=y1(ix1)
          iy2=y1(ix2)
          b1=float(iy1-iy2)/float(ix1-ix2)
          a1=float(iy1)-b1*float(ix1)        !y1=a1+b1*x1

          ix1=int(float(N2)/4.0)
          ix2=int(3.0*float(N2)/4.0)
          iy1=y2(ix1)
          iy2=y2(ix2)
          b2=float(iy1-iy2)/float(ix1-ix2)
          a2=float(iy1)-b2*float(ix1)        !y2=a2+b2*x2

          if ((abs(b1).le.0.00001).and.(abs(b2).le.0.00001)) then
cj          type*,'lines are // to hori. direction, no cross point'
            return
          end if


          if (abs(b1-b2).le.0.00001) then
cj          type*,'lines are parallel, no cross point'
            return
          end if
```

```
          xf=(a1-a2)/(b2-b1)
          x=int(xf)
          yf=a1+b1*xf
          y=int(yf)
        end if


        if ((con1.eq.1).and.(con2.eq.2)) then
          ix1=int(float(N1)/4.0)
          ix2=int(3.0*float(N1)/4.0)
          iy1=y1(ix1)
          iy2=y1(ix2)
          b1=float(iy1-iy2)/float(ix1-ix2)
          a1=float(iy1)-b1*float(ix1)        !y1=a1+b1*x1

          iy1=int(float(N2)/4.0)
          iy2=int(3.0*float(N2)/4.0)
          ix1=x2(iy1)
          ix2=x2(iy2)
          b2=float(ix1-ix2)/float(iy1-iy2)
          a2=float(ix1)-b2*float(iy1)        !x2=a2+b2*y2

          if ((abs(b1).le.0.00001).and.(abs(b2).le.0.00001)) then
cj          type*,'these two lines are perpendicular'
            x=int(a2)
            y=int(a1)
            return
          end if

          yf=(a1+a2*b1)/(1-b1*b2)
          y=int(yf)
          xf=a2+b2*yf
          x=int(xf)
        end if

        if ((con1.eq.2).and.(con2.eq.1)) then
          iy1=int(float(N1)/4.0)
          iy2=int(3.0*float(N1)/4.0)
          ix1=x1(iy1)
          ix2=x1(iy2)
          b1=float(ix1-ix2)/float(iy1-iy2)
          a1=float(ix1)-b1*float(iy1)        !x1=a1+b1*y1

          ix1=int(float(N2)/4.0)
          ix2=int(3.0*float(N2)/4.0)
          iy1=y2(ix1)
          iy2=y2(ix2)
          b2=float(iy1-iy2)/float(ix1-ix2)
          a2=float(iy1)-b2*float(ix1)        !y2=a2+b2*x2

          if ((abs(b1).le.0.00001).and.(abs(b2).le.0.00001)) then
cj          type*,'these two lines are perpendicular'
            x=int(a1)
            y=int(a2)
            return
          end if

          xf=(a1+a2*b1)/(1-b1*b2)
          x=int(xf)
          yf=a2+b2*xf
          y=int(yf)
        end if
```

```
        if ((con1.eq.2).and.(con2.eq.2)) then
          iy1=int(float(N1)/4.0)
          iy2=int(3.0*float(N1)/4.0)
          ix1=x1(iy1)
          ix2=x1(iy2)
          b1=float(ix1-ix2)/float(iy1-iy2)
          a1=float(ix1)-b1*float(iy1)        !x1=a1+b1*y1

          iy1=int(float(N2)/4.0)
          iy2=int(3.0*float(N2)/4.0)
          ix1=x2(iy1)
          ix2=x2(iy2)
          b2=float(ix1-ix2)/float(iy1-iy2)
          a2=float(ix1)-b2*float(iy1)        !x2=a2+b2*y2

          if ((abs(b1).le.0.00001).and.(abs(b2).le.0.00001)) then
cj          type*,'lines are // to vert. direction, no cross point'
            return
          end if

          if (abs(b1-b2).le.0.00001) then
cj          type*,'lines are parallel, no cross point'
            return
          end if

          yf=(a1-a2)/(b2-b1)
          y=int(yf)
          xf=a1+b1*yf
          x=int(xf)
        end if

        return
        end
c*******************************************************
c
c     Program to detect primary top position of lung
c
c     Explain: input chest images must be 1000 x 1215
c     or 1000 x 1000, which are 14" X 17" or 14" X 14"
c     chest films digitized by Konica laser digitizer,
c     using reduce factor of 2; 1 PV = 0.35 mm;
c
c     Name: top_lung_sub.for
c
c     History: March 10,1992
c              modified 3/21/93
c              modified 4/10/93
c              modified 4/16/93
c              modified by Xin-Wei for anysize image
c
c*********************************************************
c
c
        subroutine top_lung_sub(image,nxw,nyw,top_lung,top_image,
     &                             success)

        implicit integer*2 (i-n)

        integer*2
     &     nxw,nyw,          !input image size
     &     image(nxw,nyw)  !image buffer
        integer*4
     &     top_lung,         !primary top lung position
     $     top_image       !bottom of blank whit top;=1 if no white top
        integer*2
     &     success          !=1, success in finding top of lung;
                             !=0, fail in finding top of lung;

        integer*2 top_image2   ! When calling top_nowhite_fd, since expects
integer*2
        integer*2
     &     posi(1215),       !work array
     &     top_search_end,  !work parameter
     &     prof_max_posi,   !work parameter
     %     counter           ! counter number for looping

        integer*2 top_lung2,top_lung3  ! Because we pass toplung to profile_im_sub,
which expects integer*2

        real
     &     prof_vert(1215),prof_vert_smo(1215),
     &     prof_hori(1215),
     &     fd(1215),pixsmm,
     $     profSD(1215), profTOP(1215)

c ------- modify to determine ----------------------
c
c       (1): lung close to image edge;
c       (2): a narrow and tilt white edge at top
c
c       using two narrow hori profiles,
c       width 5 pixel (500 size image) seperate 5
c       pixels? 2/12/96
c
c---------------------------------------------------


        real p_h_temp1(1215),sp_h_temp1(1215),
     &       p_h_temp2(1215),sp_h_temp2(1215)


c**** program begin *****

        top_image = 1       ! THIS IS A DEFAULT!! I THINK THE SUBROUTINE COULD GO
ALL THE WAY THROUGH WITHOUT SETTING top_image!! -- Roger

        pixsmm=350.0/float(nxw) ! pixel size in mm;

        top_search_end=int(3.0*float(nyw)/8.0+0.5)

c       inc_width=int(float(nyw)/200.)  ! width of the profile
c       if (inc_width.eq.0) inc_width=1
        inc_width=5

        is=int(float(nxw)/7.0+0.5)    ! check from 1/7 of image width
        ie=int(6.0*float(is)+0.5)     ! check end at 6/7 of image width

c       inc=nxw/90
c       inc_odd=2*(int(inc/2.0))+1      !nxw=1000, inc_odd=11;
        inc_odd=11


c --- 2/12/97: set initial two profiles for check lung location ---

        index=1
```

```
        itx1=1
        itx2=nxw

        ity1=1
        ity2=4

        call profile_im_sub(image,p_h_temp1,nxw,nyw,itx1,itx2,
     $                      ity1,ity2,index) ! original profile

        call prof_smo_sub(p_h_temp1,itx2,itx2,inc_odd,sp_h_temp1)!smoothed one

        open(unit=33,file='p_h_temp1')
           do i=itx1,itx2
             write(33,*)i, sp_h_temp1(i)
           end do
        close(33)

        ity1=8
        ity2=12

        call profile_im_sub(image,p_h_temp2,nxw,nyw,itx1,itx2,
     $                      ity1,ity2,index) ! original profile

        call prof_smo_sub(p_h_temp2,itx2,itx2,inc_odd,sp_h_temp2)!smoothed one

        open(unit=33,file='p_h_temp2')
           do i=itx1,itx2
             write(33,*)i, sp_h_temp2(i)
           end do
        close(33)

        call parameter_horiProf(sp_h_temp2,sp_h_temp1,itx2,ie,is,
     &                  PVmax_min,ave,PVmin,sigma,cc,cd,rcc_cd)

c       write(6,*)'cc, cd, ratio(cc/cd):',cc, cd, rcc_cd
c        write(6,*)'-----------------------------------'
c       write(6,*)'max-min,ave,min,sigma:',PVmax_min,ave,PVmin,sigma
c        write(6,*)'***********************'

        if ((cd.lt.2.0) .and. (sigma.lt.120.0)) then
          top_lung=3
          success=1                                    ! 2/18/97
          go to 200
         else
          istarti=9
          ind=1
        end if

c*********************** 2/12/97 *************************************


c *** obtain standard hori profile at 3/8, in vertical of image ***

        loc_ySD=top_search_end  !location of SD hori profile

        ix1=1
        index=1
        call profile_im_sub(image,prof_hori,nxw,nyw,ix1,nxw,
     $           loc_ySD-inc_width,loc_ySD,index) ! original profile

        call prof_smo_sub(prof_hori,nxw,nxw,inc_odd,profSD)!smoothed one
```

```
c *** end of SD hori profile ***

c *** find top lung position by using vertcal prof in image center ***

        nn1=nxw/4
        nn2=3*nn1
        nn3=1
        nn4=2

        call profile_im_sub(image,prof_vert,nxw,nyw,nn1,nn2,nn3,nyw,nn4)

! vertical profile in midline arae with width of half of image width

        call prof_smo_sub(prof_vert,nyw,nyw,inc_odd,prof_vert_smo)

! 11 point simple smooth of prof_vert, smooth is moved to here


!top lung search range is within upper 3/8 of image height area


        prof_ave=0.0           !find ave PV in upper 3/8 area
        do j=istarti,top_search_end     ! 2/18/97: j=1, top_search_end
          prof_ave=prof_ave+prof_vert_smo(j)
        end do
        prof_ave=prof_ave/float(top_search_end)

        prof_max=prof_vert_smo(istarti) !find max PV in upper 3/8 area; 2/18/97
istarti=1
        prof_max_posi=1
        do j=istarti+1, top_search_end  !2/18/97: istarti=1
          if (prof_vert_smo(j).gt.prof_max) then
            prof_max=prof_vert_smo(j)
            prof_max_posi=j
          end if
        end do

        ratio_max_ave=prof_max/prof_ave  !ratio of Pmax/Pave in upper 3/8

        posi_max=pixsmm*float(prof_max_posi)

! distance of position of max profile value from top of image(in mm)

c       type*,'INFO about the vertical profile:'
c       type*,'Vp_max,Vp_ave,R', prof_max,prof_ave,ratio_max_ave
c       type*,'prof_max_posi(PV),posi_max(mm)',prof_max_posi,posi_max

c*********************************************************

c       if ((ratio_max_ave.ge.(0.01*posi_max+1.2)).and.
c     1     (posi_max.le.20.0)) then
c         if (prof_max.le.750.0) then
c           ind=3       !no BW,top lung position is a step edge
c          else
c           ind=2       !has blank white unexpo. area at top
c         end if
c        else
c         ind=1         !no blank white unexpo. area at top
c       end if
c
c       if (ind.eq.1) then                              !   eliminate on 2/18/97
c         write(6,*)' No blank white area in top lung '
c        else if (ind.eq.2) then
```

```
c         write(6,*)' Has blank white area in top lung '
c        else
c         write(6,*)' No BW in top, top lung is step edge '
c       end if

c*********************************************************************

! in chest_pack.for and chest_pack2.for vert. prof smooth (11 point)
! is at this step, i.e., afetr blank white detection.This is may not good.

c*********************************************************************

c***  loop for detect top of lung *****

        nn1=5
        nn2=1

        call fd_south_sub(prof_vert_smo,posi,fd,nyw,nyw,nn1,no_fd)

        i_start=istarti                    ! 2/18/97: i_start=1
        counter=1

100     continue
c       type*,' #### **** Loop Time: *** ###',counter

c*****************************************************

        if (ind.eq.3) then

          top_image=1

          iend=int(top_search_end/2.0)
          do i=1,no_fd
            if (posi(i).ge.iend) then
              istop=i
              go to 50
            end if
          end do

50        continue
          fd_min=fd(istarti)    ! 2/18/97: (1)
          top_lung=posi(istarti) ! 2/18/97: (1)
          do i=istarti+1,istop   ! 2/18/97: (2)
            if (fd(i).lt.fd_min) then ! for step edge using min peak
              fd_min=fd(i)             ! of FD to define the top position
              top_lung=posi(i)         ! in the range of half of the 3/8.
            end if
          end do
          go to 150

        end if

        top_image2 = top_image
        if (ind.eq.1) then
          call top_nowhite_fd(prof_vert_smo,nyw,nn1,top_search_end,
     &          i_start,itop,top_image2,posi,fd,index1)
        end if
        top_image = top_image2

        top_image2 = top_image
        if (ind.eq.2) then
          call top_white_fd(prof_vert_smo,nyw,nn1,top_search_end,
     &          i_start,itop,top_image2,posi,fd,index1)
```

```
        end if
        top_image = top_image2

        if (index1.eq.0) then
          if (counter.eq.1) then
            success=0
cj          type*,'top lung finding fail:No zero-crossing in FD'
cj          type*,'*** may need density correction ***'
cj          type*,'program output guessed top lung position'
            top_lung=int(top_search_end/2.0)
            go to 150
           else
            top_lung=itop_pre
            go to 200
          end if
        end if

c*************************************************************

        call profile_im_sub(image,prof_hori,nxw,nyw,nn2,nxw,
     &                     itop,itop+ic_width,nn2)

! check horizontal profile at detected top lung position

        call prof_smo_sub(prof_hori,nyw,nxw,inc_odd,profTOP)

! 11 point simple smooth of prof_hori


        call parameter_horiProf(profTOP,profSD,nxw,ie,is,PVmax_min,
     %                ave,PVmin,sigma,cc,cd,rcc_cd)


        sigma_cal=0.2*PVmax_min+50.0

        if (counter.eq.1) then
          if (sigma.ge.sigma_cal) then
            i_start=itop
            itop_pre=itop
            counter=counter+1
            go to 100
           else
            top_lung=itop
            success=1
            go to 200
          end if
        end if

        if (counter.gt.1) then

           if (sigma.ge.sigma_cal) then
             if (cc.gt.0.80) then
c              type*,'current det. top is in lung,top_lung=itop_pre'
               top_lung=itop_pre
               success=1
               go to 200
              else
               i_start=itop
               itop_pre=itop
               counter=counter+1
               go to 100
             end if
           end if
```

```
           if (prof_vert_smo(itop_pre).ge.prof_vert_smo(itop)) then
             top_lung=itop_pre
            else
             top_lung=itop
           end if
           success=1
           go to 200
        end if

c ******************************************************************

150     continue

        top_lung2 = top_lung
        top_lung3 = top_lung+ic_width
        call profile_im_sub(image,prof_hori,nxw,nyw,nn2,nxw,
     &                     top_lung2,top_lung3,nn2)
        top_lung = top_lung2

! check horizontal profile at detected top lung position

        call prof_smo_sub(prof_hori,nyw,nxw,inc_odd,profTOP)

! 11 point simple smooth of prof_hori


        call parameter_horiProf(profTOP,profSD,nxw,ie,is,PVmax_min,
     %                ave,PVmin,sigma,cc,cd,rcc_cd)
c*******************************************************************

200     continue


c       type*,' TOP LUNG==',top_lung

        return
        end

c***********************************************************
c
c       subroutine of find parameters from hori prof's
c
c       Name: parameter_horiProf.for
c
c       4/11/93
c***********************************************************

        subroutine parameter_horiProf(prof,profSD,nprof,ie,is,
     %          PVmax_min,ave,PVmin,sigma,cc,cd,rcc_cd)

        implicit integer*2 (i-n)

        integer*2 nprof,ie,is

        real    prof(nprof),profSD(nprof)

        real PVmax_min,ave,PVmin,sigma,cc,cd,rcc_cd
```

```
        real*8 sum, sigma2

c *** program begin ***

c *** begin of standard hori profile ***

        sum=0.0
        k=0
        do i=is,ie,20
          k=k+1
          sum=sum+profSD(i)
        end do
        aveSD=sum/float(k)

        sigma2=0.0
        k=0
        do i=is,ie,20
          k=k+1
          sigma2=sigma2+(profSD(i)-aveSD)**2
        end do
        sigmaSD=sqrt(sigma2/float(k-1))
        sigmaA=sqrt(sigma2)

        pminSD=profSD(is)
        do l=is,ie
          if (profSD(l).lt.pminSD) pminSD=profSD(l)
        end do

        ratio_sigma_aveSD=sigmaSD/aveSD
        ratio_sigma_pminSD=sigmaSD/pminSD

c       type*,'*** INFO of standard(SD) hori profile ***'
c       type*,'ave,pmin,sigma:', aveSD,pminSD,sigmaSD
c       type*,'ratio_sigma_ave(SD):',ratio_sigma_aveSD
c       type*,'ratio_sigma_pmin(SD):',ratio_sigma_pminSD

c *** end of this part ***

c *** begin of hori profile at top of lung ***

        pmax=prof(is)
        do l=is,ie
          if (prof(l).gt.pmax) pmax=prof(l)
        end do

        pmin=prof(is)
        do l=is,ie
          if (prof(l).lt.pmin) pmin=prof(l)
        end do

        PVmax_min=pmax-pmin
        PVmin=pmin


        sum=0.0
        k=0
        do i=is,ie,20
          k=k+1
          sum=sum+prof(i)
        end do
        ave=sum/float(k)

        sigma2=0.0
```

```
        k=0
        do i=is,ie,20
          k=k+1
          sigma2=sigma2+(prof(i)-ave)**2
        end do
        sigma=sqrt(sigma2/float(k-1))
        sigmaB=sqrt(sigma2)


        ratio_sigma_ave=sigma/ave
        ratio_sigma_pmin=sigma/pmin

c       type*,'*** INFO of hori profile at TOP LUNG ***'
c       type*,'ave,pmin,sigma:', ave,pmin,sigma
c       type*,'PVmax-PVmin:',PVmax_min
c       type*,'ratio_sigma_ave(Top):',ratio_sigma_ave
c       type*,'ratio_sigma_pmin(Top):',ratio_sigma_pmin

c *** *** *** cross-difference between prof and profSD *** *** ***

        cd=0.0
        do i=is,ie,20
          cd=cd+(prof(i)-profSD(i))**2
        end do
        sigmaAB=sigmaA*sigmaB
        cd=cd/sigmaAB
c       type*,'cross-diference:cd', cd

c *** *** *** end of cross-difference calculation *** *** ***

c *** *** *** cross-correlation between profSD and prof *** *** ***

        cc=0.0
        do i=is,ie,20
          cc=cc+(profSD(i)-aveSD)*(prof(i)-ave)
        end do
        cc=cc/sigmaAB
c       type*,'cross-correlation:cc', cc

c *** *** *** end of cross-correlation calculation *** *** ***

        rcc_cd=cc/cd
c       type*,'ratio of cc/cd=',rcc_cd

c *** end of this part ***

        return
        end

c**************************************************
c
c      subroutine of lung top detection for no
c      white area in up lungs
c
c      pro : smoothed vertical profile in mid up lungs
c      kkk : increatment of sobel filter(in pixel number)
c
c       index:(=0,no zero-cross if the FD)/(=1,has zero-crossing)
c
c       modified in 4/11/93
c**************************************************
        subroutine top_nowhite_fd(pro,iy,kkk,i_end,i_ori,i_top,
     &                           top_image,posi,fd,index)
```

```
        implicit integer*2 (i-n)

        integer*2 posi(iy),fd_no,top_image

        real pro(iy),fd(iy)

c******** begin of progranm ***********

        i_top=1
        top_image=1
        index=1

        call fd_south_sub(pro,posi,fd,iy,iy,kkk,fd_no) ! F.D. of pro

          do i=1,fd_no
            if (posi(i).gt.i_ori) then
              is=i
              go to 50
            end if
          end do
50        continue

c*** start search point of first 0 crossing & -slop from i_ori ****

          do i=is,fd_no-1
            if ((fd(i).ge.0.0).and.(fd(i+1).lt.0.0)) then!0 crossing & "-slop"
              grad=fd(i)-fd(i+1) !gradient at 0 cross must langer than 0.1
              if (grad.ge.0.1) then
                top=float(posi(i)+posi(i+1))/2.0
                i_top=int(top+0.5)
                go to 100
               else

                no_psign=1
                no_nsign=1

                do jp=i-1,is,-1
                  if (fd(jp).ge.0.0) then
                    no_psign=no_psign+1
                   else
                    go to 60
                  end if
                end do
60              continue

                do jn=i+2,fd_no-1
                  if (fd(jn).lt.0.0) then
                    no_nsign=no_nsign+1
                   else
                    go to 70
                  end if
                end do
70              continue
c               type*,'The grad. at 0 crossing is less than 0.1'
c               type*,'no_psign,no_nsign==',no_psign,no_nsign
                if ((no_psign.ge.8).and.(no_nsign.ge.8)) then
                  top=float(posi(i)+posi(i+1))/2.0
                  i_top=int(top+0.5)
                  go to 100
                end if

              end if
```

```
            end if
          end do
          index=0 ! no zero-crossing in FD

100       continue

          return
          end
c
c*****************************************************
c
c      subroutine of lung top detection for having
c      white area in up lungs
c
c      pro : smoothed vertical profile in mid up lungs
c      kkk : increatment of sobel filter(in pixel number)
c
c       index:(=0,no zero-crossing in FD)/(=1,has zero-crossing)
c
c       modified 4/11/93
c
c*****************************************************
c
        subroutine top_white_fd(pro,iy,kkk,i_end,i_ori,i_top,
     &                        top_image,posi,fd,index)

        implicit integer*2 (i-n)

        integer*2 posi(iy),fd_no, top_image

        real pro(iy),fd(iy)

c***** begin of program ********************

        index=1
        i_top=1

        call fd_south_sub(pro,posi,fd,iy,iy,kkk,fd_no)

c***** find blank white area range **********

        i_peak=1
        peak=pro(1)
        do i=2, i_end
          if (pro(i).gt.peak) then
            peak=pro(i)
            i_peak=i
          end if
        end do     ! find peak caused by white araes

        Ppeak=0.9*peak
        do i=i_peak, i_end
          if ((pro(i).ge.Ppeak).and.(pro(i+1).le.Ppeak)) then
            i90per=i
            go to 100
          end if
        end do     ! find position of 90% of white area peak
100     continue

        do i=i90per, i_end
          if ((pro(i).ge.pro(i+1)).and.(pro(i+2).gt.pro(i+1))) then
            is1=i+1    !is1: primary lower edge of blank white top
            go to 150
```

```
          end if
        end do
150     continue

        top_image=is1

          do i=1,fd_no
            if (posi(i).ge.is1) then
              ib1=i
              go to 200
            end if
          end do   !ib1: to avoid blank white area
200       continue


          do i=1,fd_no
            if (posi(i).gt.i_ori ) then
              ib2=i
              go to 250
            end if
          end do    !ib2: to avoid previous neck or chin position
250       continue

          if (ib2.ge.ib1) then
            is=ib2
           else
            is=ib1
          end if

c**** start search point of first 0 crossing & -slop from i_ori ***

          do i=is,fd_no-1
            if ((fd(i).ge.0.0).and.(fd(i+1).lt.0.0)) then!0 crossing & "-slop"
              grad=fd(i)-fd(i+1)!gradient must greater than 0.1 at top lung
              if (grad.ge.0.1) then
                top=float(posi(i)+posi(i+1))/2.0
                i_top=int(top+0.5)
                go to 300
               else

                no_psign=1
                no_nsign=1

                do jp=i-1,is,-1
                  if (fd(jp).ge.0.0) then
                    no_psign=no_psign+1
                   else
                    go to 260
                  end if
                end do
260             continue

                do jn=i+2,fd_no-1
                  if (fd(jn).lt.0.0) then
                    no_nsign=no_nsign+1
                   else
                    go to 270
                  end if
                end do
270 .           continue
c               type*,'The grad. at 0 crossing is less than 0.1'
c               type*,'no_psign,no_nsign==',no_psign,no_nsign
                if ((no_psign.ge.8).and.(no_nsign.ge.8)) then
```

```
                  top=float(posi(i)+posi(i+1))/2.0
                  i_top=int(top+0.5)
                  go to 300
                end if

              end if
            end if
          end do
          index=0       !no zero-crossing in the FD

300       continue

          return
          end
c*****************************************
c
c     program to mark a point by a cross
c
c     History: April 3,1992
c
c     Name: mark_cross_sub.for
c
c*****************************************
c
c
        subroutine mark_cross_sub(image,nxw,nyh,IX,IY,R1,R2)

        implicit integer*2 (i-n)

        integer*2
     &       nxw,nyh,                !buffer size
     &       image(nxw,nyh),         !image buffer
     &       IX,IY,                  !mark center point
     &       R1,                     !half size of the cross mark
     &       R2                      !half thickness of mark, R2 < R1

c**** start of the program *********

        do j=IY-R2,IY+R2
          do i=IX-R1, IX+R1
            image(i,j)=1023
          end do
        end do

        do i=IX-R2,IX+R2
          do j=IY-R1, IY+R1
            image(i,j)=1023
          end do
        end do

        return
        end
c*****************************************
c
c     program to mark a point by a circle
c
c     History: March 26,1992
c
c     Name: mark_circle_sub.for
c
c*****************************************
c
c
```

```
        subroutine mark_circle_sub(image,nxw,nyh,IX,IY,R1,R2)

        implicit integer*2 (i-n)

        integer*2
     &       nxw,nyh,              !buffer size
     &       image(nxw,nyh),       !image buffer
     &       IX,IY,                !mark center point
     &       R1,                   !radius of the circle mark
     &       R2,                   !thickness of circle mark, R2< R1
     &       RX

c**** start of the program *********

        do Rx=R1-R2, R1+R2
          do i=IX-Rx, IX+Rx
            y=float(Rx*Rx-(IX-i)*(IX-i))
            y=sqrt(y)
            j1=IY-int(y)
            j2=IY+int(y)
            image(i,j1)=1023
            image(i,j2)=1023
          end do
        end do

        return
        end
c*******************************************************************
c
c    program for calculating gradient
c    at a point; using soble filter
c    (x: in east direction; y: in south
c        direction)
c
c    Name: gradient_east_south_sub.for
c
c    History: May 21, 1992
c
c*******************************************************************

        subroutine gradient_east_south_sub(image,nxw,nyh,ix,iy,Nx,Ny,
     $                                   gradient,angle)

        implicit integer*2 (i-n)

        integer*2
     &      nxw,nyh,             ! buffer size
     &      image(nxw,nyh),      ! image buffer
     &      ix,iy,               ! position where gradient be calculated
     &      Nx,Ny                ! mask size for gradient calculation


        gx=0.0
        gy=0.0

        do j=iy-1,iy+1
          gx=image(ix+Nx,j)-image(ix-Nx,j)+gx
        end do
        gx=gx/(2.0*3.0*Nx)

        do i=ix-1,ix+1
          gy=image(ix,iy+Ny)-image(ix,iy-Ny)+gy
        end do
```

```
        gy=gy/(2.0*3.0*Ny)

        gradient=sqrt(gx*gx+gy*gy)

        if ((gy.eq.0.0).and.(gx.eq.0.0)) then
          angle=0.0
          go to 10
        end if

cj      angle=atan2d(gy,gx)

        angle=atan2(gy,gx)/3.1415926*180.0


10      continue

        return
        end
c*******************************************************************
c
c    program for calculating gradient
c    at a point; using soble filter
c    (x: in west direction; y: in south
c        direction)
c
c    Name: gradient_west_south_sub.for
c
c    History: May 21, 1992
c
c*******************************************************************

        subroutine gradient_west_south_sub(image,nxw,nyh,ix,iy,Nx,Ny,
     $                                  gx,gy,gradient,angle)

        implicit integer*2 (i-n)

        integer*2
     &      nxw,nyh,             ! buffer size
     &      image(nxw,nyh),      ! image buffer
     &      ix,iy,               ! position where gradient be calculated
     &      Nx,Ny                ! mask size for gradient calculation


        gx=0.0
        gy=0.0

        do j=iy-1,iy+1
          gx=image(ix-Nx,j)-image(ix+Nx,j)+gx
        end do
        gx=gx/(2.0*3.0*Nx)

        do i=ix-1,ix+1
          gy=image(ix,iy+Ny)-image(ix,iy-Ny)+gy
        end do
        gy=gy/(2.0*3.0*Ny)

        gradient=sqrt(gx*gx+gy*gy)

        if ((gy.eq.0.0).and.(gx.eq.0.0)) then
          angle=0.0
          go to 10
        end if
```

```
cj      angle=atan2d(gy,gx)

        angle=atan2(gy,gx)/3.1415926*180.0

10      continue

        return
        end

C**********************************************************************
C
C            SUBROUTINE FOR REARANGING One Dimension ARRAY of IPX
C
C            FROM Small VALUE TO Large VALUE, and IPY follow the IPX
c
c            whatever the order of the value
C
C            NAME: REarrange_S_L.FOR
C
C**********************************************************************
C
C
      SUBROUTINE REarrange_S_L(IPX,IPY,N,No_real)
C
C
      IMPLICIT integer*2 (I-N)

      integer*2 IPX(N),IPY(N),
     &          No_real          ! real dimension of the array
C
C
      DO I=1,No_real
        J=I
        IMIN=IPX(J)
        IP=J
        DO K=J,No_real
          IF (IPX(K).LT.IMIN) THEN
            IMIN=IPX(K)
            IP=K
          END IF
        END DO
        ITX=IPX(J)
        ITY=IPY(J)
        IPX(J)=IPX(IP)
        IPY(J)=IPY(IP)
        IPX(IP)=ITX
        IPY(IP)=ITY
      END DO
C
C
      RETURN
      END

C**********************************************************************
C
C            SUBROUTINE FOR REARANGING One Dimension ARRAY of IPX
C
C            FROM Large VALUE TO Small VALUE, and IPY follow the IPX
c
c            whatever the order of the value
C
C            NAME: REarrange_L_S.FOR
```

```
C
C**********************************************************************
C
C
      SUBROUTINE REarrange_L_S(IPX,IPY,N,No_real)
C
C
      IMPLICIT integer*2 (I-N)

      integer*2 IPX(N),IPY(N),
     &          No_real          ! real dimension of the array
C
C
      DO I=1,No_real
        J=I
        IMAX=IPX(J)
        IP=J
        DO K=J,No_real
          IF (IPX(K).GT.IMAX) THEN
            IMAX=IPX(K)
            IP=K
          END IF
        END DO
        ITX=IPX(J)
        ITY=IPY(J)
        IPX(J)=IPX(IP)
        IPY(J)=IPY(IP)
        IPX(IP)=ITX
        IPY(IP)=ITY
      END DO
C
C
      RETURN
      END
C**********************************************************************
C
C           SUBROUTINE to mark the ROI POSITION in the image
C
C           SUBROUTINE NAME: ROI_mark_sub.FOR;
C
C**********************************************************************
C
C
      SUBROUTINE ROI_mark_sub(Image,nxw,nyh,LX1,LX2,LY1,LY2)
C
C
      IMPLICIT integer*2 (I-N)

      integer*2 Image(nxw,nyh)   ! image buffer
      integer*2 lx1,lx2,ly1,ly2   ! ROI position in the image
C
C
      DO IX=LX1,LX2
        Image(IX,LY1)=1023
        Image(IX,LY2)=1023
      END DO
      DO IY=LY1,LY2
        Image(LX1,IY)=1023
        Image(LX2,IY)=1023
      END DO
C
C
      RETURN
      END
c******************************************************************************
c
c  program of calculating histogram in a ROI in a image
c
c       (The image is 10 bits)
c
c  name: Histogram_ROI_sub.for;
c
c******************************************************************************
c
c
      subroutine Histogram_ROI_sub(Image,nxw,nyh,x1,x2,y1,y2,bin,
     &                             No,Hist_pixel,Hist_freq)
c
c
        implicit integer*2 (i-n)

        integer*2 Image(nxw,nyh)  ! image buffer
        integer*2 x1,x2,y1,y2,     ! ROI position in the image
     &            bin,             ! histogram bin size
     &            Hist_pixel(1024),! pixel value
     &            No               ! number of histogram data after bin comprass

        real
     &            Hist_freq(1024)  ! occurance frequency of pixel value
c******************************************************************************

      integer total, sum,
     &        hist(0:1023), hist_sum(1024)


c *** initialize histogram ***

      do i=0,1023
        hist(i)=0
      end do

c *** end of this part ***

c********* obtain histogram in the ROI **********

      do j=y1,y2
        do i=x1,x2
          hist(image(i,j))=hist(image(i,j))+1
        end do
      end do

c********* end of this part *************


c******* consider bin compression ************

      fNo=1023./float(bin)
      No=int(fNo)+1
      incre=int(float(bin)/2.+0.5)

      i1=0
      i2=bin-1
      sum=0

      do i=1,No
```

```
        do j=i1,i2
          sum=hist(j)+sum
        end do
        Hist_pixel(i)=i1+incre
        if(Hist_pixel(i).gt.1023) Hist_pixel(i)=1023
        hist_sum(i)=sum
        sum=0
        i1=i1+bin
        i2=i2+bin
        if (i2.GT.1023) i2=1023
      end do

c *** end of this part ***

c**** get total pixel number of the image

      total=0
      do i=1,No
        total=total+hist_sum(i)
      end do

c***** normalize histogram in term of percentage ****

      ftotal_bin=float(total*bin)
      do i=1,No
        Hist_freq(i)=100.*(float(hist_sum(i))/ftotal_bin)
      end do

c *** end of this part ***


        return
        end
```

```
!==============================================================================
      subroutine
chs_sub(iwork_orig,iwork_2_orig,mszx_orig,mszy_orig,ires_r_x,ires_r_y,
     &         ires_l_x,ires_l_y,num_1,num_2,
     &         iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &         ix1_rule3,ith_avgpix)
!==============================================================================
!       Originaly coded by Takayuki Ishida in 1996
!       Modified by Takayuki Ishida, Jan.5, 1998
!       Add comments by Takayuki Ishida, Jan.22, 1998
!
!       Modified by Qiang Li, enable it to process images with any matrix size
!       Oct. 30, 1998
!------------------------------------------------------------------------------
        implicit   integer*4 (i-n)
!------------------------------------------------------------------------------
!       Function of this subroutine is to determine the cardiac edges for
!       lung segmentation of the current image.
!
!       (1) The current image (586 x 586 matrix size) is reduced to 65 x 65
!           matrix size.
!       (2) Candidates of the heart edges are extracted initially by histogram
!           analysis of the low-resolution image.
!       (3) The feature analysis with edge gradient and edge orientation was
!           applied for determination of the heart edges in the low-resolution
image.
!       (4) Then, the final heart edges on the original image are obtained
!           based on edge detection.
!------------------------------------------------------------------------------

        parameter  (mszx=586)            ! Marix size of the current image.(column)
        parameter  (mszy=586)            ! Marix size of the current image.(line)
        parameter  (mszx_500=586)                  ! Marix size of the current
image.(column)
        parameter  (mszy_500=586)                  ! Marix size of the current
image.(line)
        parameter  (mszx_sl=65)                    ! Marix size of the low-resolution
image
                                                   ! for detection of the cardiac
edge.(column)
        parameter  (mszy_sl=65)                    ! Marix size of the low-resolution
image
                                                   ! for detection of the cardiac
edge.(line)


        integer*2  iwork_orig(mszx_orig,mszy_orig)              ! current image
        integer*2  iwork_2_orig(mszx_orig,mszy_orig)

        integer*2  iwork(mszx_500,mszy_500)             ! current image
        integer*2  iwork_2(mszx_500,mszy_500)
        integer*2  iwt(586,586)
        integer*2  img_65(mszx_sl,mszy_sl),img_500(mszx_500,mszy_500)
        integer*2  sb(mszx_sl,mszy_sl),cardio(mszx_sl,mszy_sl)
        integer*2  nn0,nn2,nn3,nn4,ierr,ierr2
        integer*2  nn,ma,mopt,naut,ihtwk,iwx,iwy
        integer*4  ihist(0:1023),cand(mszx_sl,mszy_sl),jdist(5,100)
        integer*4  cand2(mszx_sl,mszy_sl)
        integer*4  isub_r(100),isub_l(100)
        integer*4  ires_r_x(100),ires_r_y(100)
        integer*4  ires_l_x(100),ires_l_y(100)
        integer*4  i_r(2,100),i_l(2,100)
        real*4     res_r_x(100),res_r_y(100)
```

```
        real*4     res_l_x(100),res_l_y(100)
        real*4     sbdir(mszx_sl,mszy_sl),lung_area,heart_area
        real*4     coef(21),coef2(21),ihst16x(1024),ihst16y(1024)
        real*4     cand_x_r(100),cand_y_r(100),cand_x_l(100),
     &             cand_y_l(100)
        integer*2
     &            Rribcage(2,1215),RribcageNo, ! R ribcage output
     &            Lribcage(2,1215),LribcageNo, ! L ribcage output
     %            Rdiaph(2,1215),RdiaphNo,   ! R diaphragm output
     %            Ldiaph(2,1215),LdiaphNo    ! L diaphragm output
        integer*2 feature(50) ! landmarks of chest, from ribcage detection
        real*4    cf_rib_r(21),cf_rib_l(21)
        real*4    cf_dia_r(21),cf_dia_l(21)
        integer*4 ix,iy,ix2,iy2,ierr3
        integer*2  isbdir(mszx_sl,mszy_sl),Rindex,Lindex
!------------------------------------------------------------------------------
        integer*4 ix1_rule2,iy1_rule2   ! Upper-right location of mediastinum
        integer*4 ix2_rule2,iy2_rule2   ! Upper-left of left cardiac edge
        integer*4 ix1_rule3(2,mszy_500) ! x-locations of cardiac edges
                                        ! ix1_rule3(1,SIZE586) => right cardiac
x-location
                                        ! ix1_rule3(2,SIZE586) => left cardiac
x-location
        integer*4 iy1_rule1             ! bottom of right cardiac edge
!
!       (ix1_rule2,iy1_rule2) --------> x        |
!                                       |        |
!                                       |        |
!                                       |        |
!                                       |        x <-------- (ix2_rule2,iy2_rule2)
!                                      /          \
!                                     |            \
!                                     |             \
!                                     |              |
!                iy1_rule1  --------> x              |
!
!------------------------------------------------------------------------------

c       ----------------------------------------------------------
c       enlarge/reduce the size of image to 586*586
c       because the subroutine is designed to process this image size
c       ----------------------------------------------------------
        call Image_Scaling(iwork_orig, mszx_orig, mszy_orig, iwork, mszx_500,
mszy_500)
        call Image_Scaling(iwork_2_orig, mszx_orig, mszy_orig, iwork_2, mszx_500,
mszy_500)

c       ----------------------------------------------------------
c       SET PARAMETERS
c       ----------------------------------------------------------
        ratio=0.1        ! Threshold level for histogram area

        isbx=mszx_sl
        isby=mszy_sl

        i_sty=16         ! starting point of y for center box
        i_endy=48        ! end point of y for center box

        isbint=1         ! Interval of Sobel operation

        icenter=0        ! Centering indicator 1
        iredoct=0        ! Centering indicator 2
```

```
c       ----------------------------------------------------------
c       copy img to 500x500 matrix
c       ----------------------------------------------------------
        do j=1,mszy_500
          do i=1,mszx_500
               img_500(i,j)=iwork(i,j)
               iwt(i,j)=0
          end do
        end do
c       ----------------------------------------------------------
c       Initialization
c       ----------------------------------------------------------
           do i=1,1024
              ihst16x(i)=0
              ihst16y(i)=0
           end do

           do i=1,100
              cand_x_r(i)=0.0
              cand_y_r(i)=0.0
              cand_x_l(i)=0.0
              cand_y_l(i)=0.0
              isub_r(i)=0
              isub_l(i)=0
              ires_r_x(i)=0
              ires_r_y(i)=0
              ires_l_x(i)=0
              ires_l_y(i)=0
              res_r_x(i)=0.0
              res_r_y(i)=0.0
              res_l_x(i)=0.0
              res_l_y(i)=0.0
           end do

           do i=1,21
              cf_dia_r(i)=0
              cf_dia_l(i)=0
              cf_rib_r(i)=0
              cf_rib_l(i)=0
              coef(i)=0
              coef2(i)=0
           end do

           do j=1,1215
             do i=1,2
              Rdiaph(i,j)=0
              Ldiaph(i,j)=0
             end do
           end do

              RdiaphNo=0
              LdiaphNo=0

           do i=1,100
              do j=1,5
                jdist(j,i)=0
              end do
           end do

           do j = 1, isby
             do i = 1, isbx
                img_65(i,j)=0
                sb(i,j)=0
```

```
                sbdir(i,j)=0.0
             end do
           end do
c       ----------------------------------------------------------
c       Ribcage and Diaphragm detection
c       ----------------------------------------------------------
        call lung_boundary(img_500,mszx_500,mszy_500,feature,
     &          Rribcage,RribcageNo,Lribcage,LribcageNo,
     &          Rdiaph,RdiaphNo,Ldiaph,LdiaphNo,
     &          Rindex,Lindex,cf_rib_r,cf_rib_l,
     &          cf_dia_r,cf_dia_l)
c          do i=1,10
c             write(*,333) cf_rib_r(i),cf_rib_l(i)
c          end do
c 333      format (2f10.4)

c        call diaphragm_detection (img_500,mszx_500,mszy_500,feature,
c     %          Rdiaph,RdiaphNo,Ldiaph,LdiaphNo,cf_dia_r,cf_dia_l)

c       ----------------------------------------------------------
c       Diaphragm detection Failure indication
c       ----------------------------------------------------------
  5        min_y_l = 10000
           do i=1,LdiaphNo
              if(Ldiaph(2,i).lt.min_y_l) min_y_l=Ldiaph(2,i)
           end do
           if (min_y_l.eq.10000) then
               write(*,*) 'Failed in Left diaphragm detection'
               min_y_l=feature(2)
           end if
c       ----------------------------------------------------------
c       Determine lungbottom and midline
c       ----------------------------------------------------------
           lungbottom=int(float(feature(2))/9.0+0.5)
           lungbottom_l=int(float(min_y_l)/9.0+0.5)
           if (icenter.eq.0) midline=32
           if (icenter.eq.1) midline=int(float(feature(8))/9.0+0.5)
           if (abs(int(float(feature(8))/9.0+0.5)-32).ge.5)
     &                       midline=int(float(feature(8))/9.0+0.5)

        i_stx=midline-16         ! starting point of x for center box
        i_endx=midline+16        ! end point of x for center box
c       ----------------------------------------------------------
c       REDUCE IMAGE USING AVERAGE    (63 x 76)
c       ----------------------------------------------------------
        istep=9
        icty=0
        do j=1,(mszy_500/istep)*istep,istep
           icty=icty+1
           ictx=0
           do i=1,(mszx_500/istep)*istep,istep
            ictx=ictx+1
            itotal=0
            do m=j,j+istep-1
                do l=i,i+istep-1
                   itotal=itotal+img_500(l,m)
                end do
            end do
            avgres=float(itotal)/float(istep**2)
            img_65(ictx,icty)=avgres
           end do
        end do
c       ----------------------------------------------------------
```

```
c       MAKE HISTOGRAM
c       ------------------------------------------------------------
        do ll=0,1023
            ihist(ll)=0
        end do

        do j = i_sty, i_endy
          do i = i_stx, i_endx
             ih=img_65(i,j)
             ihist(ih)=ihist(ih)+1
          end do
        end do

        ihct=0

        do ll=0,1023,16
          iwk=ihist(ll)+ihist(ll+1)+ihist(ll+2)+ihist(ll+3)+
     &    ihist(ll+4)+ihist(ll+5)+ihist(ll+6)+ihist(ll+7)+ihist(ll+8)+
     &    ihist(ll+9)+ihist(ll+10)+ihist(ll+11)+ihist(ll+12)+
     &    ihist(ll+13)+ihist(ll+14)+ihist(ll+15)
             ihct=ihct+1
             ihst16x(ihct)=float(ll)
             ihst16y(ihct)=float(iwk)
        end do
c      ------------------------------------------------------------
c       SMOOTHING HISTOGRAM (original histogram bin=16)
c      ------------------------------------------------------------
        nn=ihct+2
        ma=5
        do i=1,2
           call smooth(ihst16y,nn,ma)
        end do
c      ------------------------------------------------------------
c       SEARCH TOP OF MOUNTAINS
c      ------------------------------------------------------------
        do i=1,ihct
         if (int(ihst16y(i)).gt.0) then
            ist= i
            goto 998
         end if
        end do

 998    do i=ihct,1,-1
         if (int(ihst16y(i)).gt.0) then
            ien= i
            goto 999
         end if
        end do

 999    idivide=(ien-ist)/2+ist

        max_half1=-10000
        do ll=1,idivide
         if (ihst16y(ll).gt.max_half1) then
            max_half1=ihst16y(ll)
            istart=ll
         end if
        end do
        max_half2=-10000
        do ll=idivide+1,ihct
         if (ihst16y(ll).gt.max_half2) then
            max_half2=ihst16y(ll)
            iend=ll
```

```
          end if
         end do
c       ------------------------------------------------------------
c        SEARCH THRESHOLD VALUE FOR HEART EDGE CANDIDATES
c       ------------------------------------------------------------
          iminimumx=10000
          iminimumy=10000
          do ll=istart+1,iend-1
           if (ihst16y(ll).le.iminimumy) then
              iminimumx=ihst16x(ll)
              ist=ll
              istorg=ist
              iminimumy=ihst16y(ll)
           end if
          end do

c         write(*,*) 'p1,p2,low,high,tani',max_half1,
c      &       max_half2,istart*16,iend*16,ist*16
        ith_avgpix=ist*16
c       ------------------------------------------------------------
c        FEATURE OF HISTOGRAM
c       ------------------------------------------------------------
cc         write(*,*) 'feat',float(max_half1)/float(max_half2),
cc      &          float(ist-istart)/float(iend-istart),truth(ip)
          if (float(ist-istart)/float(iend-istart).lt.0.34.and.
     &       float(max_half1)/float(max_half2).lt.0.7) then
          ist=(iend-istart)/2+istart
          end if
c       ------------------------------------------------------------
c        determine ratio of regions LUNG vs HEART
c       ------------------------------------------------------------
c          do ll=1,ist
c             lung_area=lung_area+ihst16y(ll)
c          end do
c             heart_area=((33.*39.)-lung_area)/(33.*39.)
c             lung_area=lung_area/(33.*39.)
c             write(*,*) heart_area*100,lung_area*100
c       ------------------------------------------------------------
c        write histogram
c       ------------------------------------------------------------
c          write(*,*) ' '
c          do ll=2,ihct
c            write(*,*) ihst16x(ll), ihst16y(ll)
c          end do
c       ------------------------------------------------------------
c        determine range for detection of heart edge candidates
c       ------------------------------------------------------------
 6500     continue
          iarea=ihst16y(ist)
          wk1=float(i_endx-i_stx+1)*float(i_endy-i_sty+1)

          do k=1,10
           iarea=iarea+ihst16y(ist+k)+ihst16y(ist-k)
           if (float(iarea).gt.wk1*ratio) then
           istart=ihst16x(ist-k)
           iend=ihst16x(ist+k)
c           write(*,*) k, ihst16y(ist)
           goto 5000
           end if
          end do
c       ------------------------------------------------------------
c         HEART EDGE CANDIDATES BY USING TWO THRESHOLDS
c       ------------------------------------------------------------
```

```
 5000   continue
        do j = 1, isby
           do i = 1, isbx
              cand(i,j)=0
              cand2(i,j)=0
           end do
        end do

        do j = i_sty, i_endy
          do i = i_stx, i_endx
             if (img_65(i,j).ge.istart) cand(i,j)=1
             if (img_65(i,j).ge.iend)   cand2(i,j)=1
           end do
        end do
c       ------------------------------------------------------------
c       [B] MORPHOLOGICAL FILTERING
c       ------------------------------------------------------------
        nco=8
c        call epct2(cand,cand,isbx,isby,2,isbx-1,2,isby-1,nco,0,1)
c        call epct2(cand,cand,isbx,isby,2,isbx-1,2,isby-1,nco,1,1)

c        call epct2(cand2,cand2,isbx,isby,2,isbx-1,2,isby-1,nco,0,1)
c        call epct2(cand2,cand2,isbx,isby,2,isbx-1,2,isby-1,nco,1,1)

        call epct2(cand,cand,isbx,isby,3,isbx-2,3,isby-2,nco,0,1)
        call epct2(cand,cand,isbx,isby,3,isbx-2,3,isby-2,nco,1,1)

        call epct2(cand2,cand2,isbx,isby,3,isbx-2,3,isby-2,nco,0,1)
        call epct2(cand2,cand2,isbx,isby,3,isbx-2,3,isby-2,nco,1,1)
c       ------------------------------------------------------------
c       SUBTRACT BETWEEN TWO BINARY IMAGES
c       ------------------------------------------------------------
        do j = i_sty+1, i_endy-1
          do i = i_stx+1, i_endx-1
             cand(i,j)=cand(i,j)+cand2(i,j)
             if (j.lt.lungbottom) then
             if (cand(i,j).eq.2) then
                if (cand(i,j-1).eq.0.or.
     &              cand(i-1,j).eq.0.or.
     &              cand(i+1,j).eq.0.or.
     &              cand(i,j+1).eq.0) then
                    cand(i,j)=1
                end if
             end if
             end if
             if (j.gt.lungbottom+4) cand(i,j)=0
           end do
        end do
c      ------------------------------------------------------------
c       Sobel operation      3 X 3
c      ------------------------------------------------------------
        call sobel2(img_65,sb,sbdir,isbx,isby,isbint)
c        call writeimage(sb,'sb.img',isbx,isby)
c        do j = 1,isby
c          do i = 1, isbx
c                isbdir(i,j)=int(sbdir(i,j)*3)
c          end do
c        end do
c        call writeimage(isbdir,'sbdir.img',isbx,isby)
c      ------------------------------------------------------------
c       FEATURE ANALYSIS
c      ------------------------------------------------------------
        do j = i_sty+1, i_endy-1
```

```
          do i = i_stx+1, i_endx-1
           if (cand(i,j).eq.1) then
c       ------------------------------------------------------------
c       Edge gradient  (output from Sobel filter)
c       ------------------------------------------------------------
           if (sb(i,j).lt.300) then
                 cand(i,j)=0
           end if
c       ------------------------------------------------------------
c       Edge orientation (output from Sobel filter)
c       ------------------------------------------------------------
          if (i.lt.midline) then
             if (sbdir(i,j).gt.20.and.sbdir(i,j).le.320) then
                   cand(i,j)=0
             end if
          else
             if (sbdir(i,j).gt.0.and.sbdir(i,j).le.160) then
                   cand(i,j)=0
             end if
             if (sbdir(i,j).gt.240.and.sbdir(i,j).le.360) then
                   cand(i,j)=0
             end if
          end if

          else
             cand(i,j)=0
           end if
           end do
        end do


c        do j = i_sty+1, i_endy-1
c            write(*,555) (cand(i,j),i=i_stx+1,i_endx-1)
c        end do
c 555    format (63i1)
c       ------------------------------------------------------------
c       end of feature analysis
c       ------------------------------------------------------------

c       ------------------------------------------------------------
c       write features
c       ------------------------------------------------------------
c        icou=0
c        do j = i_sty+1, i_endy-1
c          do i = i_stx+1, i_endx-1
c             if (cand(i,j).eq.1) then
c              icou=icou+1
c           write(*,1212) icou,i,img_65(i,j),sb(i,j),int(sbdir(i,j))
c             end if
c           end do
c        end do
c 1212   format('www',i6,i6,i6,i6,i6)
c 1213   format(4i6)

c      ------------------------------------------------------------
c       fitting for Right and Left heart lines
c      ------------------------------------------------------------
        number_r=0
        number_l=0
c      ******************************
c       FITTING FOR RIGHT
c      ******************************
        do j = i_sty+1, i_endy-1
```

```
          do i = i_stx+1, midline-1
            if (cand(i,j).eq.1) then
               number_r=number_r+1
               cand_x_r(number_r)=float(i)
               cand_y_r(number_r)=float(j)
            end if
           end do
        end do

          nn0=number_r
          nn2=4
          nn3=3
          nn4=0
          ierr=0
        call kofitc2(cand_y_r,cand_x_r,nn0,nn2,coef,nn3,nn4,ierr)
c      *****************************
c         min & max of Y location
c      *****************************
          min_y_r= 1000
          max_y_r=-1000
          do i = 1, number_r
           if(int(cand_y_r(i)).gt.max_y_r)
     &                max_y_r=int(cand_y_r(i))
           if(int(cand_y_r(i)).lt.min_y_r)
     &                min_y_r=int(cand_y_r(i))
          end do
c      *****************************
c        ESTIMATE FOR RIGHT
c      *****************************
        number_r=0
        do i = min_y_r, max_y_r
            number_r=number_r+1
            iwy=i
          call polyfitc_integer(iwy,coef,nn3,iwx)
            cand_x_r(number_r)=float(iwx)
            cand_y_r(number_r)=float(i)
        end do
c      *****************************
c         max X
c      *****************************
         max_x_r=int(cand_x_r(number_r))
c      *****************************
c        FITTING FOR LEFT
c      *****************************
        do j = i_sty+1, i_endy-1
           do i = midline+1, i_endx-1
            if (cand(i,j).eq.1) then
               number_l=number_l+1
               cand_x_l(number_l)=float(i)
               cand_y_l(number_l)=float(j)
            end if
           end do
        end do
          nn0=number_l
          nn2=4
          nn3=3
          nn4=0
          ierr2=0
        call kofitc2(cand_y_l,cand_x_l,nn0,nn2,coef2,nn3,nn4,ierr2)
        if (number_r.le.11.or.number_l.le.11) then
             iredoct=iredoct+1
             if(iredoct.eq.1) then
c             write(*,*) 'Too small number of data for fitting : redo'
```

```
             ist=istorg
             goto 6500
             end if
        end if
c      *****************************
c         min & max
c      *****************************
          min_y_l=1000
          max_y_l=-1000
          do i = 1, number_l
           if(int(cand_y_l(i)).gt.max_y_l)
     &                max_y_l=int(cand_y_l(i))
           if(int(cand_y_l(i)).lt.min_y_l)
     &                min_y_l=int(cand_y_l(i))
          end do
c      *****************************
c        ESTIMATE FOR LEFT
c      *****************************
        number_l=0
        do i = min_y_l, max_y_l
            number_l=number_l+1
            iwy=i
          call polyfitc_integer(iwy,coef2,nn3,iwx)
            cand_x_l(number_l)=float(iwx)
            cand_y_l(number_l)=float(i)
        end do
c      *****************************
c         max X
c      *****************************
           max_x_l=int(cand_x_l(number_l))

        do i=1,number_l
              img_65(int(cand_x_l(i)),int(cand_y_l(i)))=1023
        end do
        do i=1,number_r
              img_65(int(cand_x_r(i)),int(cand_y_r(i)))=1023
        end do
c        call writeimage(img_65,'img65.img',isbx,isby)

c          write(*,*) 'iend; ',iend
c      -----------------------------------------------------------
c       EXPAND to Bottom
c      -----------------------------------------------------------
          icou_r=0
          icou_l=0
          if (max_y_r.gt.i_endy-3.or.max_y_l.gt.i_endy-3) then
c          write(*,*) 'through rule #1;  expand to bottom'
          ipara1=0
          ipara2=2
          ipara3=0
          ipara4=2
          irx=max_x_r
          iry=max_y_r
          icou_r= number_r-ipara1
          next_irx= int(cand_x_r(icou_r))
          do j=iry+ipara1,lungbottom
             imax=-10000
             do i=next_irx-ipara2, next_irx+ipara4
                if (sbdir(i,j).ge.0.and.sbdir(i,j).le.20.or.
     &                  sbdir(i,j).ge.320.and.sbdir(i,j).le.360) then
               i_org=img_65(i,j-1)+img_65(i,j)+img_65(i,j+1)
               i_next=img_65(i+1,j-1)+img_65(i+1,j)+img_65(i+1,j+1)
                   isub=(i_next-i_org)/3
```

```
                   if(isub.ge.imax) then
                     imax=isub
                     next_irx=i
                   end if
                end if
             end do
             if (imax.eq.-10000) goto 1199
                cand_x_r(icou_r)=float(next_irx)
                cand_y_r(icou_r)=float(j)
                icou_r=icou_r+1
          end do
 1199     continue

          ipara1=0
          ipara2=2
          ipara3=0
          ipara4=2
          ilx=max_x_l
          ily=max_y_l
          icou_l= number_l-ipara3
          next_ilx= int(cand_x_l(icou_l))
          do j=ily+ipara3,lungbottom
             imin=10000
c                if(img_65(next_ilx,j).gt.iend.or.sb(next_ilx,j).
c     &                      lt.400) then
                if(img_65(next_ilx,j).gt.iend) then
                            ipara2=6
                            ipara4=0
                else
                            ipara2=2
                            ipara4=2
                end if
             do i=next_ilx-ipara4, next_ilx+ipara2
                  iwy=j*9
                  nn3=4
                  call polyfitc_integer(iwy,cf_rib_l,nn3,iwx)
                  if(i*9.gt.iwx) goto 44
                if (sbdir(i,j).ge.150.and.sbdir(i,j).le.250) then
               i_org=img_65(i,j-1)+img_65(i,j)+img_65(i,j+1)
               i_next=img_65(i+1,j-1)+img_65(i+1,j)+img_65(i+1,j+1)
                   isub=(i_next-i_org)/3
                   if(isub.le.imin) then
                     imin=isub
                     next_ilx=i
                   end if
                end if
 44             continue
             end do
             if (imin.eq.10000) goto 1200
                cand_x_l(icou_l)=float(next_ilx)
                cand_y_l(icou_l)=float(j)
                icou_l=icou_l+1
          end do
 1200     continue
          goto 5500
          end if
c      -----------------------------------------------------------
c       EXPAND to Right
c      -----------------------------------------------------------
          if (max_x_r.lt.i_stx+3.and.max_y_r.lt.i_endy-3) then
c          write(*,*) 'through rule #2;  expand to right'
          ipara1=0
          ipara2=2
```

```
          ipara3=0
          ipara4=2
          irx=max_x_r
          iry=max_y_r
          icou_r= number_r-ipara1
          next_irx= int(cand_x_r(icou_r))
          do j=iry+ipara1,lungbottom
             imax=-10000
             do i=next_irx-ipara2, next_irx+ipara4
                if (sbdir(i,j).ge.0.and.sbdir(i,j).le.20.or.
     &                  sbdir(i,j).ge.320.and.sbdir(i,j).le.360) then
               i_org=img_65(i,j-1)+img_65(i,j)+img_65(i,j+1)
               i_next=img_65(i+1,j-1)+img_65(i+1,j)+img_65(i+1,j+1)
                   isub=(i_next-i_org)/3
                   if(isub.ge.imax) then
                     imax=isub
                     next_irx=i
                   end if
                end if
             end do
             if (imax.eq.-10000) goto 1201
                cand_x_r(icou_r)=float(next_irx)
                cand_y_r(icou_r)=float(j)
                icou_r=icou_r+1
          end do
 1201     continue
          end if
c      -----------------------------------------------------------
c       EXPAND to Left
c      -----------------------------------------------------------
          if (max_x_l.gt.i_endx-3.and.max_y_l.lt.i_endy-3) then
c          write(*,*) 'through rule #3;  expand to left'
          ipara1=0
          ipara2=2
          ipara3=0
          ipara4=2
          ilx=max_x_l
          ily=max_y_l
          icou_l= number_l-ipara3
          next_ilx= int(cand_x_l(icou_l))
          do j=ily+ipara3,lungbottom
             imin=10000
c                if(img_65(next_ilx,j).gt.iend.or.sb(next_ilx,j).
c     &                      lt.400) then
                if(img_65(next_ilx,j).gt.iend) then
                            ipara2=6
                            ipara4=0
                else
                            ipara2=2
                            ipara4=2
                end if
             do i=next_ilx-ipara4, next_ilx+ipara2
                  iwy=j*9
                  nn3=4
                  call polyfitc_integer(iwy,cf_rib_l,nn3,iwx)
                  if(i*9.gt.iwx) goto 45
                if (sbdir(i,j).ge.150.and.sbdir(i,j).le.250) then
               i_org=img_65(i,j-1)+img_65(i,j)+img_65(i,j+1)
               i_next=img_65(i+1,j-1)+img_65(i+1,j)+img_65(i+1,j+1)
                   isub=(i_next-i_org)/3
                   if(isub.le.imin) then
                     imin=isub
                     next_ilx=i
                  end if
               end if
 45              continue
             end do
             if (imin.eq.10000) goto 1202
                cand_x_l(icou_l)=float(next_ilx)
                cand_y_l(icou_l)=float(j)
                icou_l=icou_l+1
          end do
 1202     continue
          end if
c       ----------------------------------------------------------
c        Redo right
c       ----------------------------------------------------------
 5500     continue
          icou_r=icou_r-1
          icou_l=icou_l-1
c       ******************************
c         Renewed  max R
c       ******************************
          iiir=max(icou_r,number_r)
          max_y_r=int(cand_y_r(iiir))
          if (lungbottom-max_y_r.ge.6) then
c          write(*,*) 'through rule #4;  search heart edge R'
          ipara1=0
          ipara2=2
          ipara3=0
          ipara4=2
          ihalf=8
          icou_r=iiir-ihalf
          irx=int(cand_x_r(icou_r))
          iry=int(cand_y_r(icou_r))
          do kk=icou_r,iiir
              cand_x_r(kk)=0.0
          end do
          next_irx= irx
          do j=iry+ipara1,lungbottom
             imax=-10000
             do i=next_irx-ipara2, next_irx+ipara4
                if (sbdir(i,j).ge.0.and.sbdir(i,j).le.20.or.
     &                sbdir(i,j).ge.320.and.sbdir(i,j).le.360) then
               i_org=img_65(i,j-1)+img_65(i,j)+img_65(i,j+1)
               i_next=img_65(i+1,j-1)+img_65(i+1,j)+img_65(i+1,j+1)
               isub=(i_next-i_org)/3
               if(isub.ge.imax) then
                     imax=isub
                     next_irx=i
                  end if
                end if
             end do
                if (imax.eq.-10000) goto 7777
                cand_x_r(icou_r)=float(next_irx)
                cand_y_r(icou_r)=float(j)
                icou_r=icou_r+1
          end do
 7777     continue
          end if
c       ******************************
c         Renewed  max L
c      ******************************
          iiil=max(icou_l,number_l)
          max_y_l=int(cand_y_l(iiil))
          if (lungbottom-max_y_l.ge.6) then
```

```
c          write(*,*) 'through rule #5;  search heart edge L'
          ipara1=0
          ipara2=2
          ipara3=0
          ipara4=2
          ihalf=0
          icou_l=iiil-ihalf
          ilx=int(cand_x_l(icou_l))
          ily=int(cand_y_l(icou_l))
          do kk=icou_l,lungbottom
               cand_x_l(kk)=0.0
          end do
          next_ilx= ilx
          do j=ily+ipara1,lungbottom
             imin=10000
c                if(img_65(next_ilx,j).gt.iend.or.sb(next_ilx,j).
c     &                     lt.400) then
               if(img_65(next_ilx,j).gt.iend) then
                           ipara2=6
                           ipara4=0
               else
                           ipara2=2
                           ipara4=2
               end if
             do i=next_ilx-ipara4, next_ilx+ipara2
                 iwy=j*9
                 nn3=4
                 call polyfitc_integer(iwy,cf_rib_l,nn3,iwx)
                 if(i*9.gt.iwx) goto 46
               if (sbdir(i,j).ge.150.and.sbdir(i,j).le.250) then
              i_org=img_65(i,j-1)+img_65(i,j)+img_65(i,j+1)
              i_next=img_65(i+1,j-1)+img_65(i+1,j)+img_65(i+1,j+1)
              isub=(i_next-i_org)/3
              if(isub.le.imin) then
                    imin=isub
                    next_ilx=i
                  end if
               end if
 46             continue
             end do
             if (imin.eq.10000) goto 7778
                cand_x_l(icou_l)=float(next_ilx)
                cand_y_l(icou_l)=float(j)
                icou_l=icou_l+1
          end do
 7778     continue
          end if

c       ******************************
c         Renewed  max R again
c       ******************************
          icou_r=icou_r-1
          iiir=max(icou_r,number_r)
        if (cand_x_r(iiir-10)-cand_x_r(iiir-1).ge.4) then
c          write(*,*) 'through rule #6;  search heart edge R'
          ihalf=8
          icou_r=iiir-ihalf
          irx=int(cand_x_r(icou_r))
          iry=int(cand_y_r(icou_r))
          do kk=icou_r,lungbottom
               cand_x_r(kk)=0.0
          end do
          next_irx= irx
```

```
          do j=iry,lungbottom
             imax=-10000
             do i=next_irx-2, next_irx+2
                if (sbdir(i,j).ge.0.and.sbdir(i,j).le.40.or.
     &                sbdir(i,j).ge.300.and.sbdir(i,j).le.360) then
               i_org=img_65(i,j-1)+img_65(i,j)+img_65(i,j+1)
               i_next=img_65(i+1,j-1)+img_65(i+1,j)+img_65(i+1,j+1)
               isub=(i_next-i_org)/3
               if(isub.ge.imax) then
                     imax=isub
                     next_irx=i
                  end if
                end if
             end do
             if (imax.eq.-10000) goto 6777
                cand_x_r(icou_r)=float(next_irx)
                cand_y_r(icou_r)=float(j)
                icou_r=icou_r+1
          end do
 6777     continue
        end if
c       ******************************
c         Renewed  max L again
c      ******************************
          iiil=max(icou_l,number_l)
          lungbottom=max(lungbottom,lungbottom_l)
          max_y_l=int(cand_y_l(iiil))
c          write(*,*) 'through rule #7;  search heart edge L'
          icou_l=iiil
          ilx=int(cand_x_l(icou_l))
          ily=int(cand_y_l(icou_l))
          do kk=icou_l,lungbottom
               cand_x_l(kk)=0.0
          end do
          next_ilx= ilx
          do j=ily,lungbottom
             imin=10000
c                if(img_65(next_ilx,j).gt.iend.or.sb(next_ilx,j).
c     &                     lt.400) then
               if(img_65(next_ilx,j).gt.iend) then
                           ipara2=6
                           ipara4=0
               else
                           ipara2=2
                           ipara4=2
               end if
             do i=next_ilx-ipara4, next_ilx+ipara2
                 iwy=j*9
                 nn3=4
                 call polyfitc_integer(iwy,cf_rib_l,nn3,iwx)
                 if(i*9.gt.iwx) goto 47
               if (sbdir(i,j).ge.150.and.sbdir(i,j).le.250) then
              i_org=img_65(i,j-1)+img_65(i,j)+img_65(i,j+1)
              i_next=img_65(i+1,j-1)+img_65(i+1,j)+img_65(i+1,j+1)
              isub=(i_next-i_org)/3
              if(isub.le.imin) then
                    imin=isub
                    next_ilx=i
                  end if
               end if
 47             continue
             end do
             if (imin.eq.10000) goto 7878
```

```
          if (abs(next_ilx-cand_x_l(icou_l-1)).ge.2) goto 7878
                cand_x_l(icou_l)=float(next_ilx)
                cand_y_l(icou_l)=float(j)
                icou_l=icou_l+1
          end do
 7878     continue

c       ----------------------------------------------------------
c       PRINT DETECTED CARDIAC EDGE
c       ----------------------------------------------------------
         icou_r=icou_r-1
         icou_l=icou_l-1
        do j = 1, isby
           do i = 1, isbx
               cardio(i,j)=0
           end do
        end do

          iiir=max(icou_r,number_r)
          iiil=max(icou_l,number_l)

        do i=1,iiir
            nx1=int(cand_x_r(i))
            ny1=int(cand_y_r(i))
            if(nx1.ne.0.and.ny1.ne.0) cardio(nx1,ny1)=1
        end do

        do i=1,iiil
            nx2=int(cand_x_l(i))
            ny2=int(cand_y_l(i))
            if(nx2.gt.max_rib_l) cardio(nx2,ny2)=0
            if(nx2.ne.0.and.ny2.ne.0) cardio(nx2,ny2)=2
        end do
c           do i=1,RdiaphNo
c             cardio(Rdiaph(1,i)/9,Rdiaph(2,i)/9)=3
c           end do
c           do i=1,LdiaphNo
c             cardio(Ldiaph(1,i)/9,Ldiaph(2,i)/9)=3
c           end do
c        do j=1,mszy_sl
c          write(*,111) (cardio(i,j), i=1,mszx_sl)
c        end do
c 111      format (65i1)
c       ----------------------------------------------------------
c        determine distance between lt and rt heart edge
c       ----------------------------------------------------------
           kct=0
        do j = 1, isby
           idist_rx=0
           idist_lx=0
           do i = 1, isbx
               if(cardio(i,j).eq.1) idist_rx=i
               if(cardio(i,j).eq.1) idist_ry=j
               if(cardio(i,j).eq.2) idist_lx=i
               if(cardio(i,j).eq.2) idist_ly=j
           end do
           if(idist_rx.ne.0.and.idist_lx.ne.0) then
           kct=kct+1
           idist=idist_lx-idist_rx
           jdist(1,kct)=idist
           jdist(2,kct)=idist_rx
           jdist(3,kct)=idist_ry
           jdist(4,kct)=idist_lx
           jdist(5,kct)=idist_ly
           end if
        end do

           do k=1, kct
             if (jdist(1,k).le.10) then
               cardio(jdist(2,k),jdist(3,k))=3
               cardio(jdist(4,k),jdist(5,k))=4
             else
             if (jdist(1,k)-jdist(1,1).le.2) then
               cardio(jdist(2,k),jdist(3,k))=3
               cardio(jdist(4,k),jdist(5,k))=4
             else
               goto 7
             end if
             end if
           end do

  7     do j = 1, isby
            if(j.le.jdist(3,1).or.j.le.jdist(5,1)) then
               do i = 1, isbx
                 if(cardio(i,j).eq.1) cardio(i,j)=3
                 if(cardio(i,j).eq.2) cardio(i,j)=4
               end do
            end if
        end do

        do k=1,100
              i_r(1,k)=0
              i_r(2,k)=0
              i_l(1,k)=0
              i_l(2,k)=0
        end do

        num_1=0
        num_2=0
        do j = 1, isby
           do i = 1, isbx
                if(cardio(i,j).eq.1) then
                      num_1=num_1+1
                      ires_r_x(num_1)=i
                      ires_r_y(num_1)=j
                      i_r(1,num_1)=i
                      i_r(2,num_1)=j
                 end if
                if(cardio(i,j).eq.2) then
                      num_2=num_2+1
                      ires_l_x(num_2)=i
                      ires_l_y(num_2)=j
                      i_l(1,num_2)=i
                      i_l(2,num_2)=j
                end if
           end do
        end do

c -------------------------------------------------------
c  Do if number of data point is over 10.      6/25/99
c                                         by Taka Ishida
c -------------------------------------------------------
         if (num_2.gt.10) then
         do l=1,5
           if (i_l(1,num_2)-i_l(1,num_2-l).gt.5) then
           num_2=num_2-1
```

```
c           write(*,*) '----------------------minus l',l
           ll=l
           end if
         end do
        end if

c              write(*,*) 'num_1,num_2',num_1,num_2
        if (icenter.eq.1) goto 888
        if (num_1.lt.10.or.num_2.lt.10) then
c             write(*,*) 'Too small number; centering : redo'
              icenter=1
             goto 5
       end if
888    continue

        icou_r=0
        number_r=0
        max_x_r=10000
        max_y_r=-10000
        icou_l=0
        number_l=0
        max_x_l=10000
        max_y_l=-10000

           do i=1,LdiaphNo
c                ilocx=int(float(Ldiaph(1,i))/2.+0.5)
c                ilocy=int(float(Ldiaph(2,i))/2.+0.5)
                ilocx=Ldiaph(1,i)
                ilocy=Ldiaph(2,i)
               iwork_2(ilocx,ilocy)=1020
           end do
           do i=1,RdiaphNo
c                ilocx=int(float(Rdiaph(1,i))/2.+0.5)
c                ilocy=int(float(Rdiaph(2,i))/2.+0.5)
                ilocx=Rdiaph(1,i)
                ilocy=Rdiaph(2,i)
               iwork_2(ilocx,ilocy)=1020
           end do

           do i=1,num_1
                  iresx=ires_r_x(i)
                  iresy=ires_r_y(i)
            do m=(iresx-1)*9-8,(iresx-1)*9+20
                  i_org=0
                  i_next=0
                  isub=0
c                  img_500((iresx-1)*9-8,(iresy-1)*9+4)=1020
c                  img_500((iresx-1)*9+20,(iresy-1)*9+4)=1020
               do k=(iresy-1)*9+4-4,(iresy-1)*9+4+4
              i_org=i_org+
     &             (-(img_500(m-10,k)+2.*img_500(m-9,k)
     &            +3.*img_500(m-8,k)+4.*img_500(m-7,k)
     &            +5.*img_500(m-6,k)+6.*img_500(m-5,k)
     &            +7.*img_500(m-4,k)+8.*img_500(m-3,k)
     &            +9.*img_500(m-2,k)+10.*img_500(m-1,k))
     &          +(10.*img_500(m+1,k)+9.*img_500(m+2,k)
     &            +8.*img_500(m+3,k)+7.*img_500(m+4,k)
     &            +6.*img_500(m+5,k)+5.*img_500(m+6,k)
     &            +4.*img_500(m+7,k)+3.*img_500(m+8,k)
     &            +2.*img_500(m+9,k)+  img_500(m+10,k)))/660.
               end do
                  isub_r(m)=i_org
               end do
```

```
              imax=-10000
              do ii=(iresx-1)*9-8,(iresx-1)*9+20
               if(isub_r(ii).gt.imax) then
                  imax=isub_r(ii)
                  ixx=ii
                  iyy=(iresy-1)*9+4
               end if
           end do
                 ires_r_x(i)=ixx
                 ires_r_y(i)=iyy
                 res_r_x(i)=float(ixx)
                 res_r_y(i)=float(iyy)
c               img_500(ixx,iyy)=0
c               img_500(ixx+1,iyy)=0
c               img_500(ixx,iyy+1)=0
c               img_500(ixx+1,iyy+1)=0
        end do
c           write(1,*) num_2
           do i=1,num_2
                  iresx=ires_l_x(i)
                  iresy=ires_l_y(i)
                  if (i.lt.6) then
                      ix1=-8
                      ix2=8
                   else
                      ix1=-8
                      ix2=20
                  end if
               do m=(iresx-1)*9+ix1,(iresx-1)*9+ix2
                  i_org=0
                  i_next=0
                  isub=0
c                  img_500((iresx-1)*9+ix1,(iresy-1)*9+4)=1020
c                  img_500((iresx-1)*9+ix2,(iresy-1)*9+4)=1020
               do k=(iresy-1)*9+4-4,(iresy-1)*9+4+4
              i_org=i_org+
     &             (-(img_500(m-10,k)+2.*img_500(m-9,k)
     &            +3.*img_500(m-8,k)+4.*img_500(m-7,k)
     &            +5.*img_500(m-6,k)+6.*img_500(m-5,k)
     &            +7.*img_500(m-4,k)+8.*img_500(m-3,k)
     &            +9.*img_500(m-2,k)+10.*img_500(m-1,k))
     &          +(10.*img_500(m+1,k)+9.*img_500(m+2,k)
     &            +8.*img_500(m+3,k)+7.*img_500(m+4,k)
     &            +6.*img_500(m+5,k)+5.*img_500(m+6,k)
     &            +4.*img_500(m+7,k)+3.*img_500(m+8,k)
     &            +2.*img_500(m+9,k)+  img_500(m+10,k)))/660.
               end do
                  isub_l(m)=i_org
               end do
              imin=10000
              do ii=(iresx-1)*9+ix1,(iresx-1)*9+ix2
               if(isub_l(ii).lt.imin) then
                  imin=isub_l(ii)
                  ixx=ii
                  iyy=(ires_l_y(i)-1)*9+4
               end if
              end do
                 ires_l_x(i)=ixx
                 ires_l_y(i)=iyy
                 res_l_x(i)=float(ixx)
                 res_l_y(i)=float(iyy)
c               write(1,*) ixx,iyy
c               img_500(ixx,iyy)=0
```

```
c               img_500(ixx+1,iyy)=0
c               img_500(ixx,iyy+1)=0
c               img_500(ixx+1,iyy+1)=0
        end do
c      ----------------------------------------------------------
c       fitting for Right and Left heart lines
c      ----------------------------------------------------------
c      *****************************
c       FITTING FOR RIGHT
c      *****************************
          do i=1,21
             coef(i)=0
          end do
          nn0=num_1
          nn2=4
          nn3=3
          nn4=0
          ierr=0
        call kofitc2(res_r_y,res_r_x,nn0,nn2,coef,nn3,nn4,ierr)
c      *****************************
c       ESTIMATE FOR RIGHT
c      *****************************
         irt_min=10000
        do i = 1, num_1
            iwy=ires_r_y(i)
          call polyfitc_integer(iwy,coef,nn3,iwx)
            ires_r_x(i)=iwx
            if (ires_r_x(i).lt.irt_min) irt_min=ires_r_x(i)
c            write(*,*) 'rt',iwx,iwy
        end do
c      *****************************
c       FITTING FOR LEFT
c      *****************************
          do i=1,21
             coef2(i)=0
          end do
          nn0=num_2
          nn2=4
          nn3=3
          nn4=0
          ierr2=0
        call kofitc2(res_l_y,res_l_x,nn0,nn2,coef2,nn3,nn4,ierr2)
c      *****************************
c       ESTIMATE FOR LEFT
c      *****************************
           ilt_max=-10000
        do i = 1,num_2
            iwy=ires_l_y(i)
          call polyfitc_integer(iwy,coef2,nn3,iwx)
            ires_l_x(i)=iwx
            if (ires_l_x(i).gt.ilt_max) ilt_max=ires_l_x(i)
c            write(*,*) 'lt',iwx,iwy
        end do

c        open (1,file='cardio')
c          write(1,*) num_1
           do i=2,num_1
c               write(1,*) ires_r_x(i),ires_r_y(i)
               iwork_2(ires_r_x(i),ires_r_y(i))=1020
               iwork_2(ires_r_x(i)+1,ires_r_y(i))=1020
               iwork_2(ires_r_x(i),ires_r_y(i)+1)=1020
               iwork_2(ires_r_x(i)+1,ires_r_y(i)+1)=1020
               ix=ires_r_x(i)
```

```
              iy=ires_r_y(i)
              ix2=ires_r_x(i-1)
              iy2=ires_r_y(i-1)
          call line3(iwork_2,mszx,mszy,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,mszx,mszy,ix,iy,ix2,iy2,ierr3)
          end do
c          write(1,*) num_2
          do i=2,num_2
c              write(1,*) ires_l_x(i),ires_l_y(i)
              iwork_2(ires_l_x(i),ires_l_y(i))=1020
              iwork_2(ires_l_x(i)+1,ires_l_y(i))=1020
              iwork_2(ires_l_x(i),ires_l_y(i)+1)=1020
              iwork_2(ires_l_x(i)+1,ires_l_y(i)+1)=1020
              ix=ires_l_x(i)
              iy=ires_l_y(i)
              ix2=ires_l_x(i-1)
              iy2=ires_l_y(i-1)
          call line3(iwork_2,mszx,mszy,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,mszx,mszy,ix,iy,ix2,iy2,ierr3)
          end do
              ix=ires_r_x(1)
              iy=ires_r_y(1)
              ix2=ires_r_x(1)
              iy2=1
          call line3(iwork_2,mszx,mszy,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,mszx,mszy,ix,iy,ix2,iy2,ierr3)
              ix=ires_l_x(1)
              iy=ires_l_y(1)
              ix2=ires_l_x(1)
              iy2=1
          call line3(iwork_2,mszx,mszy,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,mszx,mszy,ix,iy,ix2,iy2,ierr3)
           if(ires_r_y(num_1).gt.ires_l_y(num_2)) then
              ix=ires_l_x(num_2)
              iy=ires_l_y(num_2)
              ix2=ires_l_x(num_2)
              iy2=ires_r_y(num_1)
          call line3(iwork_2,mszx,mszy,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,mszx,mszy,ix,iy,ix2,iy2,ierr3)
           end if

           do j=1,mszy
               ix1_rule3(1,j)=0
               ix1_rule3(2,j)=0
           end do
c            iy1_rule1=min(ires_r_y(num_1),ires_l_y(num_2))
           iy1_rule1=ires_r_y(num_1)
           ix1_rule2=ires_r_x(1)
           iy1_rule2=1
           ix2_rule2=ires_l_x(1)
           iy2_rule2=max(ires_r_y(1),ires_l_y(1))
           do j=iy2_rule2,iy1_rule1
              do i=1,mszx
                 if(iwt(i,j).gt.0) then
                     ix1_rule3(1,j)=i
                     goto 7577
                 end if
              end do
 7577         do i2=mszx,1,-1
                 if(iwt(i2,j).gt.0) then
                     ix1_rule3(2,j)=i2
                     goto 7578
                 end if
```

```
              end do
 7578       end do

c          write(*,*) iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2
c            do j=iy2_rule2,iy1_rule1
c                write(*,*) j,ix1_rule3(1,j),ix1_rule3(2,j)
c            end do
              ix=1
              iy=ires_r_y(num_1)
              ix2=ires_r_x(num_1)
              iy2=ires_r_y(num_1)
          call line3(iwork_2,mszx,mszy,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,mszx,mszy,ix,iy,ix2,iy2,ierr3)
              ix=ires_l_x(num_2)
              iy=ires_r_y(num_1)
              ix2=mszx_500
              iy2=ires_r_y(num_1)
          call line3(iwork_2,mszx,mszy,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,mszx,mszy,ix,iy,ix2,iy2,ierr3)
c        close(1)
c       ---------------------------------------------------
c       write vertical flip of image
c       --------------------------------------------------
c        call writeimage(iwork_2,'chs.img',586,586)


         call Cardiac_Edge_Scaling(ires_r_x,ires_r_y,num_1,float(mszx_orig) /
float(mszx_500))
         call Cardiac_Edge_Scaling(ires_l_x,ires_l_y,num_2,float(mszx_orig) /
float(mszx_500))

         call
Cardiac_Parm_Scaling(iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &        ix1_rule3,float(mszx_orig) / float(mszx_500))

        return
        end
```

```
subroutine copy_sub(image1,image2,ncol,nlin)
implicit integer*4 (i-n)
integer*2  image1(ncol,nlin),image2(ncol,nlin)

do j=1,nlin
   do i=1,ncol
       image2(i,j)=image1(i,j)
     if(image2(i,j).lt.0) image2(i,j)=0
     if(image2(i,j).gt.1023) image2(i,j)=1023
   end do
end do
return
end
```

```
      SUBROUTINE CORR1(IP,ITPL,ISTX,ISTY,ISX,ISY,ISA,JX,JY,CMAX)
C
CS    CALL CORR1(IP,ITPL,ISTX,ISTY,ISX,ISY,ISA,JX,JY,CMAX)
C
CP    IMAGE REGISTRATION BY FINE SCAN
CP    USING NORMALIZED CROSS-CORRELATION
C
CK    REGISTRATION, CROSS-CORRELATION
C
CD          AUG    , 1979, PROGRAMMED BY K.SAKAUE
CD          JULY  3, 1980, REFORMED   BY E.UENO
CD          6/15/98, Modified by Qiang Li,
C
CA    IP(ISX,ISY)          * INPUT INTEGER IMAGE                (IN)
CA    ITPL(ISTX,ISTY)      * TEMPLATE PICTURE                   (IN)
CA    ISA(4)               * SEARCH AREA
CA                           ISA(1),ISA(2) : LEFT-UPPER COORDINATE
CA                           ISA(3),ISA(4) : RIGHT-LOWER COORDINATE
CA                                      OR -(SEARCH AREA SIZE)  (IN)
CA    JX,JY                * THE LEFT-UPPER COORDINATE OF THE
CA                           MOST SIMILAR SUBIMAGE IN "IP"      (OUT)
CA    CMAX                 * CROSS-CORRELATION AT (JX,JY)       (OUT)
C
CN   REFERENCES:  D.I.BARNEA AND H.F.SILVERMAN:"A CLASS OF
CN                ALGORITHM FOR FAST DIGITAL IMAGE REGISTRATION",
CN                IEEE TRANS. C-21,NO.2, FEB.1972.
C
C
      implicit integer*4 (i-n)
      integer*4 ITPL(ISTX,ISTY)
      integer*4 IP(ISX,ISY)
      integer*4 ISA(4)
      integer*4 JXY(4)
      real*4    CC(2)
C-----------------------------------------
CD          Assign initial value to CC(2)
CD          6/15/98, Modified by Qiang Li,
C-----------------------------------------
      CC(1) = -1E20
      CC(2) = -1E20

      STXY=ISTX*ISTY
C
CF    TPLM : AVERAGE OF "ITPL"
CF    SWP  : VARIANCE OF "ITPL"
      SWP=0E0
      TPLM=0E0
      DO 9 J=1,ISTY
      DO 9 I=1,ISTX
        SWP=SWP+FLOAT(ITPL(I,J))**2
        TPLM=TPLM+FLOAT(ITPL(I,J))
    9 CONTINUE
      TPLM=TPLM/STXY
      SWP=SWP-STXY*TPLM**2
C
CF    SEARCH AREA SETTING
      L1X=ISA(1)
      L1Y=ISA(2)
      L2X=ISA(3)
      L2Y=ISA(4)
      IF(L2X.LT.0)L2X=L1X-L2X-1
      IF(L2Y.LT.0)L2Y=L1Y-L2Y-1
C
```

```
CF    ALLOWED RANGE OF REFERENCE POINTS
      M1X=L1X
      M1Y=L1Y
      M2X=L2X-ISTX+1
      M2Y=L2Y-ISTY+1
C
      DO 20 J=M1Y,M2Y
      DO 20 I=M1X,M2X
        IZ=I
        JZ=J
        CALL CORR0(IP,ITPL,ISTX,ISTY,ISX,ISY,IZ,JZ,TPLM,SWP,COR,JXY,CC)
        IF(COR.GE.1E0)GO TO 22
   20 CONTINUE
      JX=JXY(1)
      JY=JXY(2)
      CMAX=CC(1)
      RETURN
   22 CONTINUE
      JX=I
      JY=J
      CMAX=1E0
      RETURN
      END
      SUBROUTINE CORR0(IP,ITPL,ISTX,ISTY,ISX,ISY,IX,IY
     @               ,ATPL,SWP,COR,JXY,CC)
C
CS    CALL CORR0(IP,ITPL,ISTX,ISTY,ISX,ISY,IX,IY
CS             ,ATPL,SWP,COR,JXY,CC)
C
CP    NORMALIZED CROSS-CORRELATION
C
CK    REGISTRATION,  CROSS-CORRELATION
C
CD          AUG    , 1979, PROGRAMMED BY K.SAKAUE
CD          JULY  3, 1980, REFORMED   BY E.UENO
C
CA    IP(ISX,ISY)       * INPUT INTEGER PICTURE                       (IN)
CA    ITPL(ISTX,ISTY)   * TEMPLATE PICTURE                            (IN)
CA    IX,IY             * LEFT-UPPER COORDINATE OF THE SUBIMAGE
CA                        IN "IP"                                     (IN)
CA    ATPL              * AVERAGE OF "ITPL"                           (IN)
CA    SWP               * SUM OF THE POWER IN "ITPL"                  (IN)
CA    COR               * NORMALIZED CROSS-CORRELATION AT IX,IY      (OUT)
CA    JXY(1),JXY(2)     * THE ELEMENT CORRESPONDING TO THE HIGHEST
CA                        CROSS-CORRELATION FOUND FROM PREVIOUS
CA                        SCAN ELEMENTS                              (OUT)
CA    JXY(3),JXY(4)     * SAME AS ABOVE BUT THE SECOND HIGHEST       (OUT)
CA    CC(1)             * CROSS-CORRELATION AT JXY(1),JXY(2)         (OUT)
CA    CC(2)             * CROSS-CORRELATION AT JXY(3),JXY(4)         (OUT)
C
CN  REFERENCE:  D.I.BARNEA AND H.F.SILVERMAN:"A CLASS OF ALGORITHM
CN              FOR FAST DIGITAL IMAGE REGISTRATION",IEEE TRANS.
CN              ,VOL.C-21, NO.2, FEB.1972.
C

      implicit integer*4 (i-n)
      integer*4 ITPL(ISTX,ISTY)
      integer*4 IP(ISX,ISY)
      integer*4 JXY(4)
      real*4 CC(2)
      SS=0E0
      CP=0E0
      A2MN=0E0
   A1SUM=0E0
   STXY=ISTX*ISTY
   DO 10 J=1,ISTY
     LY=J+IY-1
     DO 11 I=1,ISTX
       LX=I+IX-1
       IF(LX.LT.1.OR.LX.GT.ISX)GO TO 12
       IF(LY.LT.1.OR.LY.GT.ISY)GO TO 12
       A2=IP(LX,LY)
       GO TO 13
12     CONTINUE
       A2=0E0
13     CONTINUE
       A1=ITPL(I,J)-ATPL
       CP=CP+A1*A2
       SS=SS+A2**2
       A2MN=A2MN+A2
       A1SUM=A1SUM+A1
11   CONTINUE
10 CONTINUE
   A2MN=A2MN/STXY
   SS=SS-STXY*A2MN**2
   CP=CP-A1SUM*A2MN
   SQ=SQRT(SS*SWP)
   IF(SQ.LE.0E0)SQ=0.000001
   COR=CP/SQ
   IF(COR.LE.CC(2))GO TO 20
   IF(COR.LE.CC(1))GO TO 21
   CC(2)=CC(1)
   JXY(3)=JXY(1)
   JXY(4)=JXY(2)
   CC(1)=COR
   JXY(1)=IX
   JXY(2)=IY
   GO TO 20
21 CONTINUE
   IF(IX.EQ.JXY(1).AND.IY.EQ.JXY(2))GO TO 20
   CC(2)=COR
   JXY(3)=IX
   JXY(4)=IY
20 CONTINUE
   RETURN
   END
```

```
      SUBROUTINE CORRR1(IP,ITPL,ISTX,ISTY,ISX,ISY,ISA,JX,JY,CMAX)
C
CS    CALL CORRR1(IP,ITPL,ISTX,ISTY,ISX,ISY,ISA,JX,JY,CMAX)
C
CP    IMAGE REGISTRATION BY FINE SCAN
CP    USING NORMALIZED CROSS-CORRELATION
C
CK    REGISTRATION, CROSS-CORRELATION
C
CD          AUG    , 1979, PROGRAMMED BY K.SAKAUE
CD          JULY  3, 1980, REFORMED   BY E.UENO
CD          6/15/98, Modified by Qiang Li,
C
CA    IP(ISX,ISY)          * INPUT INTEGER IMAGE                (IN)
CA    ITPL(ISTX,ISTY)      * TEMPLATE PICTURE                   (IN)
CA    ISA(4)               * SEARCH AREA
CA                           ISA(1),ISA(2) : LEFT-UPPER COORDINATE
CA                           ISA(3),ISA(4) : RIGHT-LOWER COORDINATE
CA                                      OR -(SEARCH AREA SIZE)  (IN)
CA    JX,JY                * THE LEFT-UPPER COORDINATE OF THE
CA                           MOST SIMILAR SUBIMAGE IN "IP"      (OUT)
CA    CMAX                 * CROSS-CORRELATION AT (JX,JY)       (OUT)
C
CN   REFERENCES:  D.I.BARNEA AND H.F.SILVERMAN:"A CLASS OF
CN                ALGORITHM FOR FAST DIGITAL IMAGE REGISTRATION",
CN                IEEE TRANS. C-21,NO.2, FEB.1972.
C
C
      implicit integer*4 (i-n)
      integer*2 ITPL(ISTX,ISTY)
      integer*2 IP(ISX,ISY)
      integer*4 ISA(4)
      integer*4 JXY(4)
      real*4    CMAX
      real*4    CC(2)
C---------------------------------------
CD          Assign initial value to CC(2)
CD          6/15/98, Modified by Qiang Li,
C---------------------------------------
      CC(1) = -1E20
      CC(2) = -1E20

      STXY=ISTX*ISTY
C
CF    TPLM : AVERAGE OF "ITPL"
CF    SWP  : VARIANCE OF "ITPL"
      SWP=0E0
      TPLM=0E0
      DO 9 J=1,ISTY
      DO 9 I=1,ISTX
        SWP=SWP+FLOAT(ITPL(I,J))**2
        TPLM=TPLM+FLOAT(ITPL(I,J))
    9 CONTINUE
      TPLM=TPLM/STXY
      SWP=SWP-STXY*TPLM**2
C
CF    SEARCH AREA SETTING
      L1X=ISA(1)
      L1Y=ISA(2)
      L2X=ISA(3)
      L2Y=ISA(4)
      IF(L2X.LT.0)L2X=L1X-L2X-1
      IF(L2Y.LT.0)L2Y=L1Y-L2Y-1
```

```
C
CF    ALLOWED RANGE OF REFERENCE POINTS
      M1X=L1X
      M1Y=L1Y
      M2X=L2X-ISTX+1
      M2Y=L2Y-ISTY+1
C
      DO 20 J=M1Y,M2Y
      DO 20 I=M1X,M2X
        IZ=I
        JZ=J
        CALL CORRR0(IP,ITPL,ISTX,ISTY,ISX,ISY,IZ,JZ,TPLM,SWP,COR,JXY,CC)
        IF(COR.GE.1E0)GO TO 22
   20 CONTINUE
      JX=JXY(1)
      JY=JXY(2)
      CMAX=CC(1)
      RETURN
   22 CONTINUE
      JX=I
      JY=J
      CMAX=1E0
      RETURN
      END


      SUBROUTINE CORRR0(IP,ITPL,ISTX,ISTY,ISX,ISY,IX,IY
     @                 ,ATPL,SWP,COR,JXY,CC)
C
CS    CALL CORRR0(IP,ITPL,ISTX,ISTY,ISX,ISY,IX,IY
CS             ,ATPL,SWP,COR,JXY,CC)
C
CP    NORMALIZED CROSS-CORRELATION
C
CK    REGISTRATION,  CROSS-CORRELATION
C
CD          AUG    , 1979, PROGRAMMED BY K.SAKAUE
CD          JULY  3, 1980, REFORMED   BY E.UENO
C
CA    IP(ISX,ISY)      * INPUT INTEGER PICTURE                   (IN)
CA    ITPL(ISTX,ISTY)  * TEMPLATE PICTURE                        (IN)
CA    IX,IY            * LEFT-UPPER COORDINATE OF THE SUBIMAGE
CA                       IN "IP"                                 (IN)
CA    ATPL             * AVERAGE OF "ITPL"                       (IN)
CA    SWP              * SUM OF THE POWER IN "ITPL"              (IN)
CA    COR              * NORMALIZED CROSS-CORRELATION AT IX,IY  (OUT)
CA    JXY(1),JXY(2)    * THE ELEMENT CORRESPONDING TO THE HIGHEST
CA                       CROSS-CORRELATION FOUND FROM PREVIOUS
CA                       SCAN ELEMENTS                          (OUT)
CA    JXY(3),JXY(4)    * SAME AS ABOVE BUT THE SECOND HIGHEST   (OUT)
CA    CC(1)            * CROSS-CORRELATION AT JXY(1),JXY(2)     (OUT)
CA    CC(2)            * CROSS-CORRELATION AT JXY(3),JXY(4)     (OUT)
C
CN  REFERENCE:  D.I.BARNEA AND H.F.SILVERMAN:"A CLASS OF ALGORITHM
CN              FOR FAST DIGITAL IMAGE REGISTRATION",IEEE TRANS.
CN              ,VOL.C-21, NO.2, FEB.1972.
C

      implicit integer*4 (i-n)
      integer*2 ITPL(ISTX,ISTY)
      integer*2 IP(ISX,ISY)
      integer*4 JXY(4)
      real*4 CC(2)
      SS=0E0
      CP=0E0
      A2MN=0E0
      A1SUM=0E0
      STXY=ISTX*ISTY
      DO 10 J=1,ISTY
        LY=J+IY-1
        DO 11 I=1,ISTX
          LX=I+IX-1
          IF(LX.LT.1.OR.LX.GT.ISX)GO TO 12
          IF(LY.LT.1.OR.LY.GT.ISY)GO TO 12
          A2=IP(LX,LY)
          GO TO 13
   12     CONTINUE
          A2=0E0
   13     CONTINUE
          A1=ITPL(I,J)-ATPL
          CP=CP+A1*A2
          SS=SS+A2**2
          A2MN=A2MN+A2
          A1SUM=A1SUM+A1
   11   CONTINUE
   10 CONTINUE
      A2MN=A2MN/STXY
      SS=SS-STXY*A2MN**2
      CP=CP-A1SUM*A2MN
      SQ=SQRT(SS*SWP)
      IF(SQ.LE.0E0)SQ=0.000001
      COR=CP/SQ
      IF(COR.LE.CC(2))GO TO 20
      IF(COR.LE.CC(1))GO TO 21
      CC(2)=CC(1)
      JXY(3)=JXY(1)
      JXY(4)=JXY(2)
      CC(1)=COR
      JXY(1)=IX
      JXY(2)=IY
      GO TO 20
   21 CONTINUE
      IF(IX.EQ.JXY(1).AND.IY.EQ.JXY(2))GO TO 20
      CC(2)=COR
      JXY(3)=IX
      JXY(4)=IY
   20 CONTINUE
      RETURN
      END
```

```
!==========================================================================
       subroutine cts_quick_y3(DefFile, PreImage, CurImage, SubImage)
!      program   CTS_QUICK
!==========================================================================
!      Quick Version of Chest Temporal Subtraction Processing
!         Incorporating an Automated Image Registration Technique
!--------------------------------------------------------------------------
!      Ver. 1.1
!      Written by Akiko Kano,  Apr.8, 1993
!      Modified by Xin-Wei Xu, Apr.22,1993   : Ribcage_Detection Version Up
!      Modified by Shige,      Apr.17,1993   : Application for FCR
!      Modified by Taka,       Aug.12,1997   : Improvement of Initial Matching
!      Modified by Taka,       Nov.20,1997   : Improvement of Warping by vector
!                                              selection
!      Modified by Taka,       Dec.4,1997    : Iterative Warping
!      Modified by Taka,       Dec.20,1997   : Iterative Warping without fitting
!      Modified by Qiang Li,   May.18,1998   : Lateral inclination correction
!      Modified by Qiang Li,   Jul.23,1998   : take image1 as previous image
!                                                  image2 as current image
!      Modified by Shige       Apr.28,1999   : Nonlinear Density Correction
!--------------------------------------------------------------------------
!      This program performs a subtraction processing between two temporally
!      sequential chest images.   Resulting subtraction image corresponds to
!      (current image)-(warped previous image)+(Offset).   Offset value is
determined
!      such that average pixel value of the subtraction image is 511.
!
!        1. Get processing parameters.
!        2. Read image data of original image pair - current chest image:Image2
!           and previous chest image:Image1.
!        3. Determine density correction factors for nonlinear density correc-
!           tion with reduced matrix sizes.
!        4. If exposure factor is 0.5 or less, apply the density correction.
!        5. Determine significance of lateral inclination based on chest mid-
!           lines for the two images.
!        6. Rotate Image1 in case the lateral inclination is significant.
!        7. Detect ribcage edges.
!        8. If exposure factor is larger than 0.5, apply the density correc-
!           tion now.
!        9. Select template ROIs on Image2 and search area ROIs on Image1 for
!           local matchings.
!       10. Perform a local matching technique based on cross-correlation method
!           to obtain shift values DX and DY for each center of template ROIs.
!           Calculations are done using reduced matrix sizes by a factor of 2.
!       11. Apply weighted two-dimensional curve fittings on DX and DY for the
!           center points of templates.  Shift values for other points inbetween
!           are calculated by linear interpolations.
!           Two subtraction images will be calculated by using differect fitted
!           shift values.  One is based on the major shift vector direction.
!           Other is based on the opposite direction.
!       12. Warp Image1 based on the fitted shift values and subtract the warped
!           Image1 from Image2.
!       13. Select good subtraction image by using HC value for each lung.
!           The Histogram Contrast value (HC value) is determined by width of the
!           histogram of the subtraction image.
!           Small HC value means a good subtraction image.
!       14. Compare HC values between the two subtraction images.  Subtraction
!           image with small HC values are employed.
!           When half lung is good but the other is poor, the shift vector direction
!           for the poor lung will be flipped.  Then the subtraction image isa
!           re-maked.
!       15. Iterative (2nd) temporal subtraction will be performed.
!           In the 2nd warping, final shift vector is determined by linear
```

```
!           interpolation of original shift vector.
!       16  Save the subtraction image.
!       17. Save the shift values, if necessary.
!
!--------------------------------------------------------------------------
!       (1) Maximum image matrix size is 512 columns x 645 lines.   It can be
!           modified by changing "MAXCOL" and "MAXLIN".
!       (2) Gray levels should be (0 - 1023).   To change this, subroutines
!           related to density correction and ribcage detection must be
!           modified.
!       (3) Gray level "0" represents Optical Density "3.0", and "1023"
!           represents "0.0".   This relationship can be inverted by giving "1"
!           as "GRAYSCALE".   To change the density range, subroutines related
!           to ribcage detection must be modified too.
!       (4) Application of nonlinear density correction is determined in tsub.def.
!--------------------------------------------------------------------------
!       Input Files    : Current Image
!                        Previous Image
!                        Density Correction LUTs        (If Necessary)
!
!       Output Files   : Subtraction Image
!                        Data File of Initial Shift-Map  (If Necessary)
!                        Data File of Fitted Shift-Map   (If Necessary)
!                        Log File
!--------------------------------------------------------------------------
!       This program calls;
!              Get_Parameters_Skip, Read_Original_Images_Skip, copy_sub,
!              Initialization, Get_Density_Correction_Factor,
!              Density_Correction_New, Leteral_Inclination_c, Rotate_Image,
!              RibCage_Detection, chs_sub, init_match, warp_rib_chs,
!              ROI_Selection, Local_Matching_Skip, Shift_Map_Fitting_Intp,
!              Warp_and_Subtraction, simple_subtraction, quanti2,
!              Save_Subtraction_ImageQ, Save_DataQ
!--------------------------------------------------------------------------
!       Compile and link command : just type  "make".
!       You need "Makefile", besides all the required source files, in order to
!       compile and link this program.
!==========================================================================
!
!--------------------------------------------------------------------------
!       PARAMETERS
!--------------------------------------------------------------------------
        implicit  none
        integer*4 MAXCOL, MAXLIN, GRAYSCALE, SKIPD, SKIPL, SKIPI
        integer*4 MAXTPS, MAXSAS, MAXPT, MAXPV, MAXOD
        integer*4 SIZE586, num_ite
        real*4    magnify

c       parameter (MAXCOL=512, MAXLIN=645)! Maximum Image Matrix Size
c       parameter (MAXCOL=600, MAXLIN=600)! Maximum Image Matrix Size
        parameter (MAXCOL=700, MAXLIN=700)! Maximum Image Matrix Size
        parameter (SIZE586=586)           ! Image Matrix Size
        parameter (GRAYSCALE=-1)          ! Inversed Grayscale ( 0 -> Darker )
        parameter (SKIPD=4)               ! Reduction Rate for Density Correction
        parameter (SKIPL=2)               ! Reduction Rate for Local Matching
        parameter (SKIPI=4)               ! Reduction Rate for Initlal Matching
        parameter (MAXTPS=64)             ! Maximum Template ROI Size
        parameter (MAXSAS=128)            ! Maximum Search Area ROI Size
        parameter (MAXPT=3000)            ! Maximum No. of ROI Pairs
        parameter (MAXPV=1023)            ! Maximum Pixel Value
        parameter (MAXOD=10)              ! Highest Order for Polynomial Fittting
        parameter (magnify=2.0)           ! Contrast factor for subtraction image
        parameter (num_ite=2)             ! Number of iteration
```

```
!--------------------------------------------------------------------------
!      POINTERS
!      (buf_FX1,buf_FY1): Fitted Shift Vector for vector including peak
!      (buf_FX2,buf_FY2): Fitted Shift Vector for vector not including peak
!      buf_image3: Subtraction image with warping (include peak of shift vector
!                  histogram) Sub1
!      buf_image5: Subtraction image with warping (not include peak of shift
!                  vector histogram) Sub2
!--------------------------------------------------------------------------
!      pointer (p_FX1, buf_FX1), (p_FY1, buf_FY1), (p_FX2, buf_FX2)
!       pointer (p_FY2, buf_FY2)
!       pointer (p_image3, buf_image3)
!       pointer (p_image5, buf_image5)
!--------------------------------------------------------------------------
!      VARIABLES(1)
!--------------------------------------------------------------------------
       integer*4 tps                   ! Template ROI Size (Pixels)
       integer*4 sas                   ! Search Area ROI Size (Pixels)
       integer*4 inc                   ! Distance Between Centers of ROIs
                                       !  (Pixels)
       integer*4 ldlimit               ! Pixel Value for Low Density Limit of
                                       !  ROI Selection
       real*4    lithres               ! Criteria of angle for Lateral Incli-
                                       !  nation Correction
       integer*4 order                 ! Order of Polynomials for Fitting
       real*4    WF(2,11)              ! LUT for Weighting Factor vs. Cross-
                                       !  Correlation Value
       integer*4 dens_corr             ! not used
                                       ! Density Cor. is determined in tsub.def
       integer*4 save_dat              ! 1 -> Save Shift-Map Data Files
                                       ! 0 -> Do Not Save Shift-Map Data Files
!--------------------------------------------------------------------------
!      VARIABLES(2)
!      IDX, IDY,tpco,saco,FITX1,FITY1,FITX2,FITY2,weight1,weight2,CCo,and anglewk
!      are formatted dimension for the 2D rectangular mapping.
!      IDX, IDY, tpc, sac, and CC are sequentially stored in the one diemsional
!      array.
!--------------------------------------------------------------------------
       integer*2 image1(MAXCOL,MAXLIN) ! Current Image Data
       integer*2 image2(MAXCOL,MAXLIN) ! Previous Image Data
       integer*2 image1_2(MAXCOL,MAXLIN)! Copy of Current Image for display
                                       ! with shift vactor
       integer*2 image2_2(MAXCOL,MAXLIN)! Copy of Previous Image for display
                                       ! with shift vactor
       integer*2 image1rw1(MAXCOL,MAXLIN)! Warped Current Image Data1
                                       ! including shift vactor peak
       integer*2 image1rw2(MAXCOL,MAXLIN)! Warped Current Image Data2
                                       ! not including shift vactor peak
       integer*2 iroiS(MAXCOL,MAXLIN)  ! Small ROI for HC determination
       integer*2 roglungimg(MAXCOL,MAXLIN)! Image that shows lung segmentation
information [ROGER]
       integer*2 rogleftribarr(MAXLIN), rogrightribarr(MAXLIN)  ! For same purpose.
       integer*2 rogribtop, rogribbot                       ! For same purpose
       integer*2 rogx1, rogx2, rogy1, rogy2                 ! For same purpose
       external   write_headerless !$pragma C(write_headerless) ! For same purpose
       real*4 rogslope
       character file1*80, file2*80    ! Filenames of Image1 and Image2
       integer*4 leng1, leng2          ! Length of Filename
       integer*4 ncol, nlin            ! Matrix Size of Image1 and 2
       integer*4 DC1, DC2              ! 1 -> Dens. Corr. is Done
                                       ! 0 -> Dens. Corr. is Not Done
       real*4    angle,anglePrev,angleCur ! Rotation Angle for Lateral Inclination
                                       !  Correction
```

```
        integer*2 blank(MAXCOL,MAXLIN)  ! 1 -> Pixel with No Image Data
                                        ! 0 -> Pixel with Image Data
        integer*2 rribcage1(2,1215)     ! Right Ribcage Points for Image1
        integer*2 lribcage1(2,1215)     ! Left  Ribcage Points for Image1
        integer*2 rribcage2(2,1215)     ! Right Ribcage Points for Image2
        integer*2 lribcage2(2,1215)     ! Left  Ribcage Points for Image2
        integer*2 ribfeature1(50)       ! Ribcage-Related Features for Image1
        integer*2 ribfeature2(50)       ! Ribcage-Related Features for Image2
        integer*2 rribcage_no1          ! No. of Right Ribcage Points for Image1
        integer*2 lribcage_no1          ! No. of Left  Ribcage Points for Image1
        integer*2 rribcage_no2          ! No. of Right Ribcage Points for Image2
        integer*2 lribcage_no2          ! No. of Left  Ribcage Points for Image2
        real*4    cf_r1(10),cf_l1(10)   ! Coefficients of Fitted Ribcage Edge for
Image1
        real*4    cf_r2(10),cf_l2(10)   ! Coefficients of Fitted Ribcage Edge for
Image2
        integer*4 sac(2,MAXPT)          ! Centers of Search Areas on Image1
        integer*4 tpc(2,MAXPT)          ! Centers of Templates on Image2
        integer*4 region1(4)            ! Smallest Rectangle Area Including sac
        integer*4 region2(4)            ! Smallest Rectangle Area Including tpc
        integer*4 number                ! Number of ROI Pairs is
                                        ! determined in the subroutine of
ROI_SELECTION.f
        integer*4 DX(MAXPT), DY(MAXPT)  ! Initial Shift Values by Cross-correlation
        integer*4 IDX(MAXPT), IDY(MAXPT)! Initial Shift Values by Cross-correlation
                                        ! formatted for the rectangler shift-value
map.
        integer*4 tpco(2,MAXPT), saco(2,MAXPT)  ! Center location for Template ROI
                                        ! and Search area ROI
                                        ! formatted for the rectangler
center-location map.
        integer*4 FITX1(MAXPT), FITY1(MAXPT)    ! Fitted Shift Values for Sub1
                                        ! formatted for the rectangler fitted
shift-value map.
        integer*4 FITX2(MAXPT), FITY2(MAXPT)    ! Fitted Shift Values for Sub2
                                        ! formatted for the rectangler fitted
shift-value map.
        real*4    weight1(MAXPT)        ! Weights for determination of Sub1
                                        ! formatted for the rectangler weight map.
        real*4    weight2(MAXPT)        ! Weights for determination of Sub2
                                        ! formatted for the rectangler weight map.
        real*4    CC(MAXPT),CCo(MAXPT)  ! Cross-Correlation Values and
                                        ! Cross-Correlation Values
                                        ! formatted for the rectangler CC map.
        real*4    angleroi(MAXPT)       ! Angle of Shift Vector for each ROI
                                        ! formatted for the rectangler angle map.
        real*4 buf_FX1(MAXCOL,MAXLIN), buf_FY1(MAXCOL,MAXLIN)          ! Buffer
Address for Fitted Shift Values
                                        ! for the majority direction
        real*4 buf_FX2(MAXCOL,MAXLIN), buf_FY2(MAXCOL,MAXLIN)          ! Buffer
Address for Fitted Shift Values
                                        ! for the minority direction
        integer*2 buf_image3(MAXCOL,MAXLIN)            ! Buffer Address for
Subtraction Image3
                                        ! for the majority direction
        integer*2 buf_image5(MAXCOL,MAXLIN)            ! Buffer Address for
Subtraction Image5
                                        ! for the minority direction

        integer   time(3)
        integer*4 id                    ! ID for Log File
        integer*4 shiftmid,ribtop       ! Global shift value
!--------------------------------------------------------------------------
```

```
!       cmax is not used in the main routine.
!-----------------------------------------------------------------------
        real*4    cmax                     ! Maximum CC value for Initial image
matching
!-----------------------------------------------------------------------
        integer*4 qmid,qtop,qbot,offset ! Midline, Toplung, Bottom lung and
                                        ! Offset for evaluation of Subtraction image
        integer*4 npx,npy                  ! MATRIX SIZE OF SHIFT-MAPS in the
rectangular
                                           ! area including all ROIs.

        integer*4 ires_r_x(100),ires_r_y(100) ! Detected location of R-cardiac edge
        integer*4 ires_l_x(100),ires_l_y(100) ! Detected location of L-cardiac edge

!-----------------------------------------------------------------------
        integer*4 ix1_rule2,iy1_rule2      ! Upper-right location of mediastinum
        integer*4 ix2_rule2,iy2_rule2      ! Upper-left of left cardiac edge
        integer*4 ix1_rule3(2,SIZE586)     ! x-locations of cardiac edges
                                           ! ix1_rule3(1,SIZE586) => right cardiac
x-location
                                           ! ix1_rule3(2,SIZE586) => left cardiac
x-location
        integer*4 iy1_rule1                ! bottom of right cardiac edge
!
!       (ix1_rule2,iy1_rule2) --------> x        |
!                                       |        |
!                                       |        |
!                                       |        |
!                                       |        x <-------- (ix2_rule2,iy2_rule2)
!                                      /          \
!                                     |            \
!                                     |             \
!                                     |              |
!                  iy1_rule1  --------> x            |
!
!-----------------------------------------------------------------------
        integer*4 num1,num2                ! Number of data for cardiac points
        integer*4 ith_avgpix               ! Threshold level (pixel value)
                                           ! for cardiac boundary

        integer*4 giveangle                ! SW, if giveangle= 1 => Majority vector
                                           !     if giveangle=-1 => Minority vector

!-----------------------------------------------------------------------
!       Histogram width of subtraction image obtained from major shift vector.
!       Histogram width is a measure of subtraction image quality.
!       The smaller histograms width corresponds to the better image quality.
!       contr1s,contr2s correspond the histogram widths for the subtraction
!       images obtained with major shift vector and minor shift vector,
respectively.
!-----------------------------------------------------------------------
        integer*4 contr1s,contr2s          ! HC values of Sub images (Small ROI)
        integer*4 contl1s,contl2s          ! HC values of Sub images (Small ROI)

c        integer*4 iswr,iswl               ! Switch vector when iswr/iswl = 1

        integer*4 iflag            ! iflag =  0 -> Select Majority
                                   ! iflag = -1 -> Select Minority
                                   ! iflag =  1 -> Switch vector in R lung
                                   ! iflag =  2 -> Switch vector in L lung
        integer*4 iloop            ! Stopper for vector flip
        integer*4 iteration        ! Number of loop for Iteration
```

```
        integer*4 i,j           ! Just used for do loop
        integer*4 ilasty        ! Used for tracking last Y position [ROGER]
        character*1  findi      ! Indicator for saving file name (Sub1 or Sub2)

        common /LOGFILE/ id
!-----------------------------------------------------------------------
!       WORK SPACE FOR SHIFT VECTOR ANALYSIS MATCHING
!       These were not used in this main routine.
!-----------------------------------------------------------------------
        integer*4 work1(MAXPT)                          ! copy of IDX
        integer*4 work2(MAXPT)                          ! copy of IDY
        real*4    work3(MAXPT)                          ! work for determination of
                                                        ! shift-vactor orientation
        real*4    work4(MAXPT)                          ! work for determination of
                                                        ! average of ROIs.
        real*4    work5(MAXPT)                          ! work for determination of
                                                        ! SD of ROIs.
        integer*4 work6(MAXPT)                          ! work for determination of
                                                        ! histogram of subtraction
image.
!-----------------------------------------------------------------------
!       WORK SPACE FOR INITIAL MATCHING
!-----------------------------------------------------------------------
        integer*4 kncol, knlin                          ! Reduced Image Size
        integer*2 wk1(MAXCOL/SKIPI,MAXLIN/SKIPI)        ! Reduced work Image
        integer*2 wk2(MAXCOL/SKIPI,MAXLIN/SKIPI)        ! Reduced work Image
        integer*2 wk3(MAXCOL/SKIPI,MAXLIN/SKIPI)        ! Reduced work Image
        integer*2 wk4(MAXCOL/SKIPI,MAXLIN/SKIPI)        ! Reduced work Image
        integer*2 wk5(MAXCOL/SKIPI,MAXLIN/SKIPI)        ! Reduced work Image
        integer*2 wk6(MAXCOL/SKIPI,MAXLIN/SKIPI)        ! Reduced work Image
        integer*2 wk7(MAXCOL/SKIPI,MAXLIN/SKIPI)        ! Reduced work Image
        integer*2 wk8(MAXCOL/SKIPI,MAXLIN/SKIPI)        ! Reduced work Image
!-----------------------------------------------------------------------
!       FUNCTIONS
!-----------------------------------------------------------------------
        integer*4 Read_Original_Images_Skip
        integer*4 ROI_Selection
        integer*4 Local_Matching_Skip
        integer   malloc
        real*4    dtime
!-----------------------------------------------------------------------
!       Given by Q. Li, used for midline detection
!---------------------------------------------------------------------
        real*4    BestShift,BestShiftPrev,BestShiftCur  ! shift value of midline in
X direction
        real*4    A, B                     ! Coefficients for "X = AY + B"

!-----------------------------------------------------------------------
!       Given by Shige (image file names and size of FCR header etc.)
!---------------------------------------------------------------------
        character       DefFile*128
        character       PreImage*128, CurImage*128, SubImage*128
        integer*4       return_value
        real*4          tarray(2), ttime
!
!-----------------------------------------------------------------------
!       GET PARAMETERS
!-----------------------------------------------------------------------
        ttime=dtime(tarray)
        call Get_Parameters_Skip( tps, sas, inc, ldlimit, lithres, order, WF,
        1          dens_corr, save_dat, SKIPL, MAXTPS, MAXSAS, MAXPV, MAXOD )
!-----------------------------------------------------------------------
!       READ CURRENT(2) AND PREVIOUS(1) IMAGES
```

```
!-----------------------------------------------------------------------
        return_value = Read_Original_Images_Skip( image2, image1,
        1               ncol, nlin, MAXCOL, MAXLIN,
        2               DefFile,
        3               PreImage,     CurImage)

!-----------------------------------------------------------------------
!       Density Correction (Application is determined in tsub.def)
!-----------------------------------------------------------------------
        call nonlinear_density_correction(image2, image1, ncol, nlin, DefFile)

!-----------------------------------------------------------------------
!       Counter for iterative temporal subtraction
!-----------------------------------------------------------------------
        iteration=0
        iflag=0
 1234   iteration=iteration+1
!-----------------------------------------------------------------------
!       Copy warped current image to Image1 as current image.
!-----------------------------------------------------------------------
        if(iteration.ge.2.and.iflag.ne.-1) then
          call copy_sub(image1rw1,image1,ncol, nlin)
        end if
        if(iteration.ge.2.and.iflag.eq.-1) then
          call copy_sub(image1rw2,image1,ncol, nlin)
        end if
!-----------------------------------------------------------------------
!       Smaller search area size will be given in the 2nd warping
!-----------------------------------------------------------------------
        if(iteration.ge.2) sas=45
!-----------------------------------------------------------------------
!       INITIALIZATION
!-----------------------------------------------------------------------
            call itime(time)
            write(id,*) 'Program Started at ',
     1          time(1), ':', time(2), ':', time(3)
            call Initialization( DC1, DC2, blank, ncol, nlin )
!-----------------------------------------------------------------------
!   COPY image1 for display of shift vector, cardiac line and Ribcage edge
!   image1_2(i,j), image2_2(i,j): Just for display purpose.
!-----------------------------------------------------------------------
          do j=1,MAXLIN
             do i=1,MAXCOL
                 image1_2(i,j)=image1(i,j)
                 image2_2(i,j)=image2(i,j)
             end do
          end do

!***********************************************************************
!***********************************************************************
!       !!!! Note that below part will not be performed for 2nd warping.
!***********************************************************************
!***********************************************************************
!-----------------------------------------------------------------------
!       ROTATE IMAGE
!-----------------------------------------------------------------------
           if(iteration.eq.1) then
c           if ( Lateral_Inclination_c( image1, image2, ncol, nlin,
c       1                     angle, lithres, image1_2, image2_2) ) then
c               call Rotate_Image( image1, ncol, nlin, angle, blank )
c               call Rotate_Image( image1_2, ncol, nlin, angle, blank )
c           end if
!-----------------------------------------------------------------------
```

```
!       Rotate image by ribcage edge based midline detection method
!       Qiang Li, 5/18/98,
!       University of Chicago
!-----------------------------------------------------------------------

c           write(*,*) 'RibCage_Detection of image1 for Image_Rotate'
            call RibCage_Detection(image1, ncol, nlin, ribfeature1,
     1      rribcage1, rribcage_no1, lribcage1, lribcage_no1,cf_r1,cf_l1)

            call Rot_Angle_By_Ribcage(image1_2, ncol, nlin, ribfeature1, rribcage1,
rribcage_no1,
    1                                lribcage1, lribcage_no1, A, B, angle,
BestShift)

            call image_shift( image1, ncol, nlin, BestShift, int(0) )
            call image_shift( image1_2, ncol, nlin, BestShift, int(0) )
            call Image_Rotate( image1, ncol, nlin, angle, int(0) )
            call Image_Rotate( image1_2, ncol, nlin, angle, int(0) )

            BestShiftPrev = BestShift
            anglePrev = angle

c           write(*,*) 'RibCage_Detection of image2 for Image_Rotate'
            call RibCage_Detection(image2, ncol, nlin, ribfeature2,
     1      rribcage2, rribcage_no2, lribcage2, lribcage_no2,cf_r2,cf_l2)

            call Rot_Angle_By_Ribcage(image2_2, ncol, nlin, ribfeature2, rribcage2,
rribcage_no2,
    1                                lribcage2, lribcage_no2, A, B, angle,
BestShift)

            call image_shift( image2, ncol, nlin, BestShift, int(0) )
            call image_shift( image2_2, ncol, nlin, BestShift, int(0) )
            call Image_Rotate( image2, ncol, nlin, angle, int(0) )
            call Image_Rotate( image2_2, ncol, nlin, angle, int(0) )

            BestShiftCur = BestShift
            angleCur = angle

            call writepreprocvals('preproc_vals\0', BestShiftPrev, anglePrev,
     1                          BestShiftCur, angleCur)

!-----------------------------------------------------------------------
!       RIBCAGE DETECTION
!-----------------------------------------------------------------------
c           write(*,*) 'RibCage_Detection of image1 for init_match'
            call RibCage_Detection(image1, ncol, nlin, ribfeature1,
       1        rribcage1, rribcage_no1, lribcage1, lribcage_no1,cf_r1,cf_l1)

c           write(*,*) 'RibCage_Detection of image2 for init_match'
            call RibCage_Detection(image2, ncol, nlin, ribfeature2,
       1        rribcage2, rribcage_no2, lribcage2, lribcage_no2,cf_r2,cf_l2)

                  do i=1,rribcage_no1
                     if(rribcage1(1,i).lt.1) rribcage1(1,i)=1
                     if(rribcage1(1,i).gt.ncol) rribcage1(1,i)=ncol
                     if(rribcage1(2,i).lt.1) rribcage1(2,i)=1
                     if(rribcage1(2,i).gt.nlin) rribcage1(2,i)=nlin
c                     write(*,*) rribcage1(1,i),rribcage1(2,i)
                  end do
c                     write(*,*) ' '
                  do i=1,lribcage_no1
                     if(lribcage1(1,i).lt.1) lribcage1(1,i)=1
```

```
                        if(lribcage1(1,i).gt.ncol) lribcage1(1,i)=ncol
                        if(lribcage1(2,i).lt.1) lribcage2(1,i)=1
                        if(lribcage1(2,i).gt.nlin) lribcage2(1,i)=nlin
c                        write(*,*) lribcage1(1,i),lribcage1(2,i)
                     end do
c                        write(*,*) ' '
                     do i=1,rribcage_no2
                        if(rribcage2(1,i).lt.1) rribcage2(1,i)=1
                        if(rribcage2(1,i).gt.ncol) rribcage2(1,i)=ncol
                        if(rribcage2(2,i).lt.1) rribcage2(2,i)=1
                        if(rribcage2(2,i).gt.nlin) rribcage2(2,i)=nlin
c                        write(*,*) rribcage2(1,i),rribcage2(2,i)
                     end do
c                        write(*,*) ' '
                     do i=1,lribcage_no2
                        if(lribcage2(1,i).lt.1) lribcage2(1,i)=1
                        if(lribcage2(1,i).gt.ncol) lribcage2(1,i)=ncol
                        if(lribcage2(2,i).lt.1) lribcage2(2,i)=1
                        if(lribcage2(2,i).gt.nlin) lribcage2(2,i)=nlin
c                        write(*,*) lribcage2(1,i),lribcage2(2,i)
                     end do

!-----------------------------------------------------------------------------
!       Create lung segmentation image [ROGER]
!-----------------------------------------------------------------------------

        do j = 1,nlin
          do i = 1,ncol
            roglungimg(nlin,ncol) = 0;
          end do
        end do

!       !!-- Now make arrays for the left and right ribcage, one element per row.
!       !!-- First, set all to 0.
        do i=1,nlin
          rogleftribarr(i) = 0
          rogrightribarr(i) = 0
        end do

!       !! Determine effective top (maximum of two) and effective bottom (minimum).
        if (rribcage1(2,1) .gt. lribcage1(2,1)) then
          rogribtop = rribcage1(2,1)
        else
          rogribtop = lribcage1(2,1)
        end if
        if (rribcage1(2,rribcage_no1) .lt. lribcage1(2,lribcage_no1))
     &       then
          rogribbot = rribcage1(2,rribcage_no1)
        else
          rogribbot = lribcage1(2,lribcage_no1)
        end if

!       !!-- Set rogrightribarr according to the rribcage1 and lribcage1
!       !!-- arrays.  If there are gaps, fill in.
        do i=1,rribcage_no1
          rogrightribarr(rribcage1(2,i)) = rribcage1(1,i)
          if ((i .lt. rribcage_no1) .and. (rribcage1(2,i+1) -
     &        rribcage1(2,i) .gt. 1)) then
            rogx1 = rribcage1(1,i)
            rogy1 = rribcage1(2,i)
            rogx2 = rribcage1(1,i+1)
            rogy2 = rribcage1(2,i+1)
            rogslope = real(rogx2 - rogx1) / (rogy2 - rogy1)
```

```
            do j=rogy1 + 1, rogy2 - 1
               rogrightribarr(j) = rogx1 + nint(rogslope * (j-rogy1))
            end do
          end if
        end do

!       !!-- Same for rogleftribarr.
        do i=1,lribcage_no1
          rogleftribarr(lribcage1(2,i)) = lribcage1(1,i)
          if ((i .lt. lribcage_no1) .and. (lribcage1(2,i+1) -
     &        lribcage1(2,i) .gt. 1)) then
            rogx1 = lribcage1(1,i)
            rogy1 = lribcage1(2,i)
            rogx2 = lribcage1(1,i+1)
            rogy2 = lribcage1(2,i+1)
            rogslope = real(rogx2 - rogx1) / (rogy2 - rogy1)
            do j=rogy1 + 1, rogy2 - 1
               rogleftribarr(j) = rogx1 + nint(rogslope * (j-rogy1))
            end do
          end if
        end do

!       !!-- Loop from top to bottom, filling in the "roglungimg" image.
        do j=rogribtop,rogribbot
          do i=rogrightribarr(j),rogleftribarr(j)
            roglungimg(i,j) = 1000
          end do
        end do

        !!-- Write the image out to "lungimage" in the current directory.
        !!-- NOTE!!! We write it byte-swapped! -+-
        call write_headerless('lungimage$$', roglungimg, MAXCOL,
     &                                ncol, nlin)

!       !!-- Now loop from top to bottom, printing.
!        write (*,*)
!        write (*,*) 'Whole array:'
!        do j=rogribtop,rogribbot
!          write (*,*) j, ' : ', rogrightribarr(j), rogleftribarr(j)
!        end do
!
!        write (*,*), 'There are ',rribcage_no1,' right ribcage points'
!        ilasty = rribcage1(2,1) - 1;
!        do i=1,rribcage_no1
!          write(*,*) rribcage1(1,i), rribcage1(2,i)
!          if (rribcage1(2,i) .ne. (ilasty+1)) then
!            write (*,*) 'Coordinate ', rribcage1(1,i), rribcage1(2,i),
!     &                   ' is out of sequence.'
!          end if
!          ilasty = rribcage1(2,i)
!        end do
!
!        write (*,*), 'There are ',lribcage_no1,' left ribcage points'
!        ilasty = lribcage1(2,1) - 1;
!        do i=1,lribcage_no1
!          write(*,*) lribcage1(1,i), lribcage1(2,i)
!          if (lribcage1(2,i) .ne. (ilasty+1)) then
!            write (*,*) 'Coordinate ', lribcage1(1,i), lribcage1(2,i),
!     &                   ' is out of sequence.'
!          end if
!          ilasty = lribcage1(2,i)
!        end do
```

```
!-----------------------------------------------------------------------------
!       Heart edge detection for current image
!-----------------------------------------------------------------------------
c           write(*,*) 'chs_sub of image1'
          call chs_sub(image1,image1_2,ncol,nlin,ires_r_x,ires_r_y,
     &         ires_l_x,ires_l_y,num1,num2,
     &         iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &         ix1_rule3,ith_avgpix)

!-----------------------------------------------------------------------------
!       INITIAL IMAGE MATCHING using cross-correlation of small blurred images
!-----------------------------------------------------------------------------
            kncol = ncol / SKIPI
            knlin = nlin / SKIPI
            call init_match(image1,image2,ncol, nlin,rribcage1,
     &           lribcage1,rribcage2,lribcage2,ribfeature1,ribfeature2,
     &           rribcage_no1,lribcage_no1,rribcage_no2,lribcage_no2,
     &           shiftmid,ribtop,cmax, kncol, knlin,
     &           wk1,wk2,wk3,wk4,wk5,wk6,wk7,wk8)
            write (*,*) 'Global shift = ' , shiftmid,ribtop

!*****************************************************************************
!*****************************************************************************
!       !!!! Note that above part will not be performed for 2nd warping
!*****************************************************************************
!*****************************************************************************
              end if
!-----------------------------------------------------------------------------
!       iterative process end
!-----------------------------------------------------------------------------
!
!-----------------------------------------------------------------------------
!       This part will be done only for 2nd warping
!       Function: Warping of Ribcade edge boundary and cardiac boundary
!                 for segmentation of warped Image1
!-----------------------------------------------------------------------------
              if(iteration.ge.2.and.iflag.ne.-1) then
          call warp_rib_chs(buf_FX1,buf_FY1,ncol, nlin,ribfeature1,
     &        rribcage1,lribcage1,rribcage_no1,lribcage_no1,
     &        ires_r_x,ires_r_y,ires_l_x,ires_l_y,num1,num2,
     &        iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &        ix1_rule3,image1_2)
              shiftmid=0
              ribtop=0
              end if
              if(iteration.ge.2.and.iflag.eq.-1) then
          call warp_rib_chs(buf_FX2,buf_FY2,ncol,nlin,ribfeature1,
     &        rribcage1,lribcage1,rribcage_no1,lribcage_no1,
     &        ires_r_x,ires_r_y,ires_l_x,ires_l_y,num1,num2,
     &        iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &        ix1_rule3,image1_2)
              shiftmid=0
              ribtop=0
              end if
!-----------------------------------------------------------------------------
!       ROI SELECTION
!-----------------------------------------------------------------------------
             if ( ROI_Selection( image2, ncol, nlin,
     &        ribfeature1, rribcage1, lribcage1,ribfeature2, rribcage2,
     &        lribcage2, rribcage_no2, lribcage_no2,sas, tps, inc,
```

```
     &        ldlimit, number, sac, tpc, region1, region2 ,
     &        shiftmid,ribtop) .eq. 1) then
!-----------------------------------------------------------------------------
!       LOCAL MATCHING
!-----------------------------------------------------------------------------
c                write(*,*) '...Local Matching.'
                 if ( Local_Matching_Skip( image1, image2, ncol, nlin, sas, tps,
     1           number, sac, tpc, region1, SKIPL,DX,DY,CC) .eq. 1) then

                     call itime(time)
                     write(id,*) 'Local Matching completed at ',
     1               time(1), ':', time(2), ':', time(3)
!-----------------------------------------------------------------------------
!       SURFACE FITTING FOR SHIFT MAP
!-----------------------------------------------------------------------------
!                  if(iteration.ge.2) then
!                    call free(p_FX1)
!                    call free(p_FY1)
!                    call free(p_FX2)
!                    call free(p_FY2)
!                    call free(p_image3)
!                    call free(p_image5)
!                   end if
!                  p_FX1 = malloc( ncol*nlin*4 )
!                  p_FY1 = malloc( ncol*nlin*4 )
!                  p_FX2 = malloc( ncol*nlin*4 )
!                  p_FY2 = malloc( ncol*nlin*4 )
!                  p_image3 = malloc( ncol*nlin*2 )
!                  p_image5 = malloc( ncol*nlin*2 )

c                  write(*,*) '...Fitting by ', order, 'th polynomials.'

                   iflag=0
                   iloop=0

1112               giveangle=1           ! Select Majority direction
                   iloop=iloop+1
              call Shift_Map_Fitting_Intp( DX, DY, CC,buf_FX1,buf_FY1,
     &             ncol, nlin, inc,number,tpc,sac,region2,WF, order,
     &             FITX1,FITY1,weight1, npx,npy,tpco,saco,IDX,IDY,CCo,
     &             angleroi,shiftmid,ribtop,
     &             iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &             ix1_rule3,ith_avgpix,image1,giveangle,
     &             ribfeature1(6),iflag,iteration,ribfeature1(1),
     &             MAXPT,work1,work2,work3,work4,work5,work6)

              if(iteration.eq.1) then
                 giveangle=-1           ! Select Minority direction
              call Shift_Map_Fitting_Intp( DX, DY, CC,buf_FX2, buf_FY2,
     &           ncol, nlin, inc,number,tpc,sac,region2,WF, order,
     &           FITX2,FITY2,weight2, npx,npy,tpco,saco,IDX,IDY,CCo,
     &           angleroi,shiftmid,ribtop,
     &           iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &           ix1_rule3,ith_avgpix,image1,giveangle,
     &           ribfeature1(6),iflag,iteration,ribfeature1(1),
     &           MAXPT,work1,work2,work3,work4,work5,work6)
              end if
!-----------------------------------------------------------------------------
!       WARPING AND SUBTRACTION
!-----------------------------------------------------------------------------
c       write(*,*)'Before WRITE ',ncol, nlin
c                    write(*,*) '...Warping of Image1.'
                     call Warp_and_Subtraction( image1, image2,
     1                          buf_image3, blank, ncol, nlin,
     2   buf_FX1, buf_FY1, GRAYSCALE, MAXPV ,offset, image1rw1,
     3   magnify)
         if(iteration.eq.1) then
                 call Warp_and_Subtraction( image1, image2,
     1                          buf_image5, blank, ncol, nlin,
     2   buf_FX2, buf_FY2, GRAYSCALE, MAXPV ,offset, image1rw2,
     3   magnify)
         end if
!-----------------------------------------------------------------------
!       Determine HC value of subtraction images
!-----------------------------------------------------------------------
            qmid = ribfeature2(6)              ! midline of image2
            qtop = ribfeature2(1)              ! toplung of image2
            qbot = ribfeature2(2)              ! bottomlung image2

            findi='1'                          ! output file indicator
         call quanti2(buf_image3,ncol, nlin, rribcage2,lribcage2,
     &        rribcage_no2,lribcage_no2,qmid,qtop,qbot,offset,
     &        iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &        ix1_rule3,contr1s,contl1s,irois,shiftmid,ribtop,findi,
     &        magnify)

         if(iteration.eq.1) then
            findi='2'
         call quanti2(buf_image5,ncol, nlin, rribcage2,lribcage2,
     &        rribcage_no2,lribcage_no2,qmid,qtop,qbot,offset,
     &        iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &        ix1_rule3,contr2s,contl2s,irois,shiftmid,ribtop,findi,
     &        magnify)
         end if
                 call itime(time)
                 write(id,*) 'Warp and Subtraction completed at ',
     1           time(1), ':', time(2), ':', time(3)
!-----------------------------------------------------------------------
!       Evaluation of subtraction image (This will be done on 1st warping only.
!-----------------------------------------------------------------------
          if(iteration.eq.1) then
              if(contr1s.le.contr2s.and.contl1s.le.contl2s) then
c                     iswr=0
c                     iswl=0
                     iflag=0
               end if
              if(contr1s.gt.contr2s.and.contl1s.gt.contl2s) then
c                     iswr=1
c                     iswl=1
                     iflag=-1
               end if
              if(contr1s.le.contr2s.and.contl1s.gt.contl2s) then
c                     iswr=0
c                     iswl=1
                     iflag=2
               end if
              if(contr1s.gt.contr2s.and.contl1s.le.contl2s) then
c                     iswr=1
c                     iswl=0
                     iflag=1
               end if
          end if

!-----------------------------------------------------------------------
!       Flip shift vectors for the right or the left lung,
!       when the histogram width of the left or the right lung obtained with
```

```
!       majority shift-vector is greater than the histogram width obtained with
!       the minority shift-vector.  Then, the minority shift-vector will be
!       employed for the correspoding lung.
!-----------------------------------------------------------------------
          if(iloop.eq.1.and.iflag.ge.1) goto 1112

      ! Write shift maps to files (Roger)

        if (iteration.eq.1) then
          if (iflag .eq. -1) then        ! Minority direction, so use shift map 2
            call writeimagefloat(buf_FX2, 'shift1.x.img\0', ncol, nlin)
            call writeimagefloat(buf_FY2, 'shift1.y.img\0', ncol, nlin)
c           write (*,*) 'ShiftMatrix1X(201,201) is ', buf_FX2(201,201)
c           write (*,*) 'ShiftMatrix1Y(201,201) is ', buf_FX2(201,201)
          else
            ! Majority direction, so use shift map 1
            call writeimagefloat(buf_FX1, 'shift1.x.img\0', ncol, nlin)
            call writeimagefloat(buf_FY1, 'shift1.y.img\0', ncol, nlin)
c           write (*,*) 'ShiftMatrix1X(201,201) is ', buf_FX1(201,201)
c           write (*,*) 'ShiftMatrix1Y(201,201) is ', buf_FX1(201,201)
          end if
        elseif (iteration .eq. 2) then
          ! Majority direction is always used.
          call writeimagefloat(buf_FX1, 'shift2.x.img\0', ncol, nlin)
          call writeimagefloat(buf_FY1, 'shift2.y.img\0', ncol, nlin)
        end if

!-----------------------------------------------------------------------
!       Goto iterative warping process.
!       In this scheme, number of warping is two.
!-----------------------------------------------------------------------
          if(iteration.lt.num_ite) goto 1234

!-----------------------------------------------------------------------
!       SAVE FINAL SUBTRACTED IMAGE
!-----------------------------------------------------------------------

               call Reverse_Value(buf_image3,ncol, nlin)

                 call Save_Subtraction_ImageQ( buf_image3, ncol, nlin,
     1                 DefFile,
     2                 PreImage, CurImage, SubImage)

                 if ( save_dat .eq. 1 ) then
                     call Save_DataQ( number, sac, tpc, inc, region2,
     1                   ncol, nlin, DX, DY, CC, buf_FX1, buf_FY1,
     2                   file1, file2, leng1, leng2, sas, tps, order )
                 end if

                 call itime(time)
                 write(id,*) 'Process completed at ',
     1           time(1), ':', time(2), ':', time(3)

!-----------------------------------------------------------------------
!       Memory release
!-----------------------------------------------------------------------
!                call free(p_FX1)
!                call free(p_FY1)
!                call free(p_FX2)
!                call free(p_FY2)
!                call free(p_image3)
!                call free(p_image5)
             else
```

```
                 write(*,*) '!!! Error in Local Matching.!!!'
                 stop
             end if
          else
             write(*,*) '!!! ROI Selection Failed.!!!'
             stop
          end if
!-----------------------------------------------------------------------
!       END
!-----------------------------------------------------------------------
        ttime=dtime(tarray)
        write(*,9123)ttime
9123    format('         CONSUMED TIME(SEC) = ',f6.1)
!       stop
        return
        end
```

```
!===============================================================================
        function  density_correction( image, ncol, nlin, DCTable )
!===============================================================================
!       Ver. 1.0
!       Written by Akiko Kano,  Mar.24, 1993
!-------------------------------------------------------------------------------
!       This function performs a nonlinear density correction and returns the
!       density-corrected image data instead of the original image data.
!
!       (1) Input image should have grayscale ranging from 0 to 1023, in which
!           "0" corresponds to high optical density and "1023" corresponds to
!           low optical density.
!===============================================================================
!       ARGUMENTS
!-------------------------------------------------------------------------------
        implicit  none
        integer*4 Density_Correction
        integer*4 ncol, nlin            ! Matrix Size of Image Data         [I]
        integer*2 image(ncol,nlin)      ! Original and Corrected Image Data[I,O]
        integer*2 DCTable(2,1024)       ! LUT for Density Correction        [I]
                                        !  DCTable(1,*):Input Pixel Value
                                        !  DCTable(2,*):Output Pixel Value
!-------------------------------------------------------------------------------
!       VARIABLES
!-------------------------------------------------------------------------------
        integer*4 C, L
!-------------------------------------------------------------------------------
!       BEGIN
!-------------------------------------------------------------------------------
        do L = 1, nlin
            do C = 1, ncol
                image(C,L) = DCTable( 2, (image(C,L)+1) )
            end do
        end do
        Density_Correction = 1
!-------------------------------------------------------------------------------
!       END
!-------------------------------------------------------------------------------
        return
        end
```

```
!========================================================================
      function Determine_Midline( image, ncol, nlin, top, bottom, primid,
     1                            mid_no, A, B )
!========================================================================
!     Ver. 1.0
!     Written by Akiko Kano,  Mar.24, 1993
!------------------------------------------------------------------------
!     This function determines midline of the chest based on smoothed hori-
!     zontal profiles.  The midline is described as "X = A * Y + B" using
!     coefficients A and B.
!
!     (1) Midline is determined by fitting points representing low density
!         points in the mediastinum between top and bottom.
!     (2) If the error in fitting is large, fitting is re-done excluding
!         points with large error.
!     (3) Returns "error" if the error is large even after the second trial.
!========================================================================
!     ARGUMENTS
!------------------------------------------------------------------------
      implicit  none
      integer*4 Determine_Midline
      integer*4 ncol, nlin            ! Matrix Size of Image Data        [I]
      integer*2 image(ncol,nlin)      ! Chest Image Data                 [I]
      integer*4 top, bottom           ! Top and Bottom of Lung           [I]
      integer*4 primid                ! Temporary midline                [I]
      integer*4 mid_no                ! Number of Points                 [O]
      real*4    A, B                  ! Coefficients for "X = AY + B"    [O]
!------------------------------------------------------------------------
!     VARIABLES
!------------------------------------------------------------------------
      real*4    PI, P_SDTHRES1, P_SDTHRES2, P_ERTHRES
      integer*4 P_HEIGHT1, P_HEIGHT2, P_WINDOW1, P_WINDOW2, MAXNO
      parameter (PI=3.141593)
      parameter (P_SDTHRES1=0.01, P_SDTHRES2=0.013, P_ERTHRES=1.5)
      parameter (P_HEIGHT1=8, P_HEIGHT2=6, P_WINDOW1=250, P_WINDOW2=100)
      parameter (MAXNO=20)
c     character str*80
      integer*4 mid(2,MAXNO)
      integer*4 clin, mcol, height, window
      integer*4 I, J
      real*4    X(MAXNO), Y(MAXNO), error(MAXNO)
      real*4    sd_err, sdthres1, sdthres2
      integer*4 id
c     integer*4 leng
      common    /LOGFILE/ id
!------------------------------------------------------------------------
!     FUNCTIONS
!------------------------------------------------------------------------
c     integer*4 Determine_Midpoint, Least_Square_1D
c     integer*4 PutOutputF, UTL$STR_PRINT, UTL$FILE_WRITE
!------------------------------------------------------------------------
♀
!------------------------------------------------------------------------
!     1. SET CRITERIA
!------------------------------------------------------------------------
      sdthres1 = nint( P_SDTHRES1 * float(ncol) )
      sdthres2 = nint( P_SDTHRES2 * float(ncol) )
!------------------------------------------------------------------------
!     2. SET SMOOTHING WIDTH FOR CALCULATION OF PROFILE
!------------------------------------------------------------------------
      do I = 1, mid_no
          clin = top + nint( float(bottom-top) * float(I) / float(mid_no+1) )
          if ( I.le.(mid_no+1)/2 ) then
```

```
              height = ( bottom - top ) / P_HEIGHT1
              window = ncol / P_WINDOW1
          else
              height = ( bottom - top ) / P_HEIGHT2
              window = ncol / P_WINDOW2
          end if
!------------------------------------------------------------------------
!     3. DETERMINE MIDPOINTS
!------------------------------------------------------------------------
          call Determine_Midpoint( image, ncol, nlin, primid, clin, mcol,
     1                                                   height, window )
          mid(1,I) = mcol
          mid(2,I) = clin
          X(I) = float(mcol)
          Y(I) = float(clin)
      end do
!------------------------------------------------------------------------
!     4. FITTING(1)
!------------------------------------------------------------------------
      call Least_Square_1D( mid_no, Y, X, A, B, error, sd_err )
!------------------------------------------------------------------------
!     5. CHECK ERROR
!------------------------------------------------------------------------
      J = 0
      do I = 1, mid_no
          if ( sd_err.le.sdthres1 .or. abs(error(I)).le.P_ERTHRES*sd_err )
     1                                                               then
              J = J + 1
              X(J) = X(I)
              Y(J) = Y(I)
          end if
      end do
!------------------------------------------------------------------------
!     6. FITTING(2)
!------------------------------------------------------------------------
      if ( J.lt.mid_no ) call Least_Square_1D( J, Y, X, A, B, error, sd_err )

c     leng = UTL$STR_PRINT( str, ' SD_err : %7.4f', sd_err )
c     call PutOutputF( '%s', str )
c     call UTL$FILE_WRITE( id, str )

      write(id,1000) ' SD_err : ', sd_err
 1000 format( A, F7.4)

      if ( sd_err.le.SDTHRES2 ) then
          Determine_Midline = 1
      else
          Determine_Midline = 0
      end if
!------------------------------------------------------------------------
!     END
!------------------------------------------------------------------------
      return
      end
♀
!========================================================================
      subroutine Determine_Midpoint( image, ncol, nlin, ccol, clin, mcol,
     1                                                 height, window )
!========================================================================
!     Ver. 1.0
!     Written by Akiko Kano,  Mar.25, 1993
!------------------------------------------------------------------------
!     This function calculates smoothed horizontal profiles around a given
```

```
!     line and detect the column with maximum profile value in the middle 1/3
!     region.
!========================================================================
!     ARGUMENTS
!------------------------------------------------------------------------
      implicit  none
c     integer*4 Determine_Midpoint
      integer*4 ncol, nlin            ! Matrix Size of Image Data        [I]
      integer*2 image(ncol,nlin)      ! Image Data                       [I]
      integer*4 ccol                  ! Temporary Midline                [I]
      integer*4 clin                  ! Center Line for Calculation      [I]
      integer*4 mcol                  ! Column No. of Midpoint           [O]
      integer*4 height                ! No. of Lines for averaging       [I]
      integer*4 window                ! Half of No. of columns for averaging
!------------------------------------------------------------------------
!     VARIABLES
!------------------------------------------------------------------------
      integer*4 width
      integer*4 C, L, CC, SC, SL, EC, EL
      integer*4 pro, minp
!------------------------------------------------------------------------
!     BEGIN
!------------------------------------------------------------------------
      width = ncol / 3
      SC = max( ccol-width/2+1, window+1 )
      EC = min( ccol+width/2, ncol-window )
      SL = max( clin-height/2+1, 1 )
      EL = min( clin+height/2, nlin )

      mcol = SC
      minp = 0
      do C = SC-window, SC+window
          do L = SL+1, EL
              minp = minp + image(C,L)
          end do
      end do

      do C = SC+1, EC
          pro = 0
          do CC = C-window, C+window
              do L = SL+1, EL
                  pro = pro + image(CC,L)
              end do
          end do
          if ( pro.gt.minp ) then
              mcol = C
              minp = pro
          end if
      end do
!------------------------------------------------------------------------
!     END
!------------------------------------------------------------------------
      return
      end
♀
!========================================================================
      subroutine Least_Square_1D( N, X, Y, A, B, error, sd_err )
!========================================================================
!     Ver. 1.0
!     Written by Akiko Kano,  Mar.25, 1993
!------------------------------------------------------------------------
!     This function performs a line fitting using least square method to
!     determine coefficients A and B for "Y = A * X + B".
```

```
!========================================================================
!     ARGUMENTS
!------------------------------------------------------------------------
      implicit  none
c     integer*4 Least_Square_1D
      integer*4 N                     ! No. of Data Sets                 [I]
      real*4    X(N), Y(N)            ! Data                             [I]
      real*4    A, B                  ! Determined Coefficients
                                      !  for "Y = AX + B"                [O]
      real*4    error(N)              ! Error                            [O]
      real*4    sd_err                ! Standard Deviation of Error      [O]
!------------------------------------------------------------------------
!     VARIABLES
!------------------------------------------------------------------------
      integer*4 I
      real*4    av_err, SX, SY, SX2, SXY
!------------------------------------------------------------------------
!     BEGIN
!------------------------------------------------------------------------
      SX = X(1)
      SY = Y(1)
      SX2 = X(1)**2
      SXY = X(1) * Y(1)

      do I = 2, N
          SX = SX + X(I)
          SY = SY + Y(I)
          SX2 = SX2 + X(I)**2
          SXY = SXY + X(I) * Y(I)
      end do

      A = ( float(N)*SXY - SX*SY ) / ( float(N)*SX2 - SX**2 )
      B = ( SY - A*SX ) / float(N)

      av_err = 0.0
      sd_err = 0.0
      do I = 1, N
          error(I) = Y(I) - A*X(I) - B
          av_err = av_err + error(I)
          sd_err = sd_err + error(I)**2
      end do
      av_err = av_err / float(N)
      sd_err = sqrt( sd_err / float(N) - av_err**2 )
!------------------------------------------------------------------------
!     END
!------------------------------------------------------------------------
      return
      end
```

```
c***********************************************************************
c       subroutine of determination of diaphragm edges
c
c       Name: diaphragm_detection
c
c       Modified from diaphragm_ribcage_mark_1k_2.for
c       requirement: ribcage detection frst
c
c       *** 9/8/93 ***
c       add rule-based left diaphragm edge detection: add a subroutine
c       called diap_Left_SP, which finds candidates of left diaphragm
c       edge starting points. The lung angle is old obtained by large
c       increatments 76 deg for Right and 108 for Left something ****
c
c       *** 9/24/93 ***
c       The update reason has describe in readme.message; check it.
c
c***********************************************************************

        subroutine diaphragm_detection (image,ncol,nlin,feature,
     %          Rdiaph,Rdiaph_No,Ldiaph,Ldiaph_No,cf_r,cf_l)

!=======================================================================
!       ARGUMENTS
!=======================================================================

        implicit integer*2 (i-n)

        integer*4 ncol,nlin         ! Matrix size of chest image[I]
        integer*2 image(ncol,nlin)  ! Inputed 1k x 1k chest image [I]
        integer*2 feature(50)       ! Landmark info. of chest image
                                    ! from ribcage detection
        integer*2 Rdiaph(2,1215)    ! Right diaphragm edges 1: x; 2 y.[O]
        integer*2 Rdiaph_No         ! No. of right diaphragm edges
        integer*2 Ldiaph(2,1215)    ! Left diaphragm edges 1: x; 2 y.[O]
        integer*2 Ldiaph_No         ! No. of left diaphragm edges

!=======================================================================
!       VARIABLES
!=======================================================================

        integer*2 nxw, nyh

        integer*2 xcoord_hori(1000),ycoord_hori(1000),
     &            xcoord_vert(1215),ycoord_vert(1215),
     &            kx(100),ky(100),kx2(100),ky2(100),
     &            count,count1,xc,yc,xf,yf,bx,byup,bylow

        integer*2 dia_r(2,100),dia_r_No

        integer*2 dia_l1(2,100),dia_l1_No,dia_l2(2,100),dia_l2_No

        integer*2 hist_pixel(1024)

        real hist_freq(1024),hist_freq_smo(1024)
        real cf_r(21),cf_l(21)
        real tempx(100),tempy(100),
     %       gradR(100),angR(100),gradL(100),angL(100)

!=======================================================================

c************ begine ****************************************************
```

```
        nxw=ncol
        nyh=nlin

        do i=1, 1024
          hist_freq(i)=0.0
          hist_freq_smo(i)=0.0
          hist_pixel(i)=0.0
        end do
        do i=1, 10
          cf_r(i)=0.0
          cf_l(i)=0.0
        end do
        do i=1, 1000
          xcoord_hori(i)=0
          ycoord_hori(i)=0
        end do
        do i=1, 1215
          xcoord_vert(i)=0
          ycoord_vert(i)=0
        end do
        do i=1, 100
          kx(i)=0
          ky(i)=0
          kx2(i)=0
          ky2(i)=0
          ky2(i)=0
          ky2(i)=0
          tempx(i)=0.0
          tempy(i)=0.0
          gradR(i)=0.0
          angR(i)=0.0
          gradL(i)=0.0
          angL(i)=0.0
          dia_r(1,i)=0
          dia_r(2,i)=0
          dia_l1(1,i)=0
          dia_l1(2,i)=0
          dia_l2(1,i)=0
          dia_l2(2,i)=0
        end do
        do i=1, 1215
          Rdiaph(1,i)=0
          Rdiaph(2,i)=0
          Ldiaph(1,i)=0
          Ldiaph(2,i)=0
        end do

c *** study rigth diaphragm edge ***

c *** find start point:
c     y: primary lungBottom;
c     X: center of cross point of lung angle line with lung bottom line
c        and minimum pv at center of lung level,previous method ***

        do i=1,1000
          xcoord_hori(i)=i
          ycoord_hori(i)=feature(2)
        end do  !! horizantol line at primary lung bottom

        index=2
        call straight_line_sub(feature(9),feature(3),feature(11),
     $       feature(5),xcoord_vert,ycoord_vert,nyh,index)
```

```
c **** straight lung angle line (through right lung angle 76 deg;corresponding
c to corrected minimum positions at 1/5 and center of lung length) ******

        lnd1=1
        lnd2=2
        call cross_p_2_line_sub(xcoord_hori,ycoord_hori,nxw,lnd1,
     $       xcoord_vert,ycoord_vert,nyh,lnd2,ixs,iys)

c ***** cross point of lung angle line and lung bottom line *****

        ixs_D_r=int(0.5*float(feature(11)+ixs))
        iys_D_r=iys

cc      write(6,*)'ixs_D_r,iys_D_r',ixs_D_r,iys_D_r ! R dia. edge SP

        feature(15)=ixs+int(0.8*float(feature(15)-ixs))!R dia. inside limit

c *** &&& end of finding start point of right diaphragm edge &&& ***

        xf=ixs_D_r
        yf=iys_D_r
        bx=10
        byup=100
        bylow=100
        count=0
        count1=0
        ind=0
        indd1=2

c *** make roi and its histogram ***

10      ix1=xf-bx
        if (ix1.le.1) ix1=2
        ix2=xf+bx
        iy1=yf-byup
        iy2=yf+bylow
        if (iy2.ge.nyh) iy2=nyh-1
        count1=count1+1

        incre=25
        if (count1.eq.1) then
cc        write(6,*)'*** %%% $$$ ### @@@ ***'
cc        write(6,*)'*** right lung histogram info. at strat point ***'
          call histogram_roi_sub(image,nxw,nyh,ix1,ix2,iy1,iy2,
     &       incre,No,hist_pixel,hist_freq)
           N_hist=1024
           inc=3
          call prof_smo_sub(hist_freq,N_hist,No,inc,hist_freq_smo)
          call histogram_analysis(hist_pixel,hist_freq_smo,No,incre,
     %                iRHPV_1stpeak)
cc        write(6,*)'*** %%% $$$ ### @@@ ***'
        end if

c *** end of this part ***

ccc****  tracing Right dia. inside edge ****ccccc

        index=0
        call Diaphragm_right(image,nxw,nyh,xf,yf,xc,yc,
     &                       bx,byup,bylow,index,indd1)
        count=count+1
        dia_r(1,count)=xc
        dia_r(2,count)=yc
```

```
        xf=xc+30  !****** trace R dia. inside edges *****
        if (xf.lt.feature(15)) then
          yf=yc
          cond=float(feature(15)-xf)/30.        ! new add in v.7
          if (cond.lt.3.0) then   ! check last two points y levels
            ind=1
           else
            ind=0
          end if                                ! new add in v. 7
          byup=30
          bylow=30
          go to 10
        end if

ccc***** tracing Right dia. outside edges ****ccccc

        xf=dia_r(1,1)-30
        yf=dia_r(2,1)
        byup=40
        bylow=60
        ind=0
        indd1=1

c *** make roi ***

20      ix1=xf-bx
        if (ix1.le.1) ix1=2
        ix2=xf+bx
        iy1=yf-byup
        iy2=yf+bylow
        if (iy2.ge.nyh) iy2=nyh-1

c *** end of this part ***

        call Diaphragm_right(image,nxw,nyh,xf,yf,xc,yc,
     &                       bx,byup,bylow,ind,indd1)
        count=count+1
        dia_r(1,count)=xc
        dia_r(2,count)=yc
        xf=xc-30  ! ***** tracing R. dia outside edges ***********
        if (xf.gt.feature(14)) then
          yf=yc
          cond=float(xf-feature(14))/30.
          if (cond.lt.2.0) then   ! check last two points y levels
            ind=1
           else
            ind=0
          end if
          go to 20
        end if
        dia_r_No=count

c *************** fit right diaphragm edges *****************

cc      write(6,*)'rearranged R. dia. edges'
        do i=1,dia_r_No
          kx(i)=dia_r(1,i)
          ky(i)=dia_r(2,i)
        end do
        kxy=100
        call REarrange_S_L(kx,ky,kxy,dia_r_No)
cc      do i=1,dia_r_No
cc        write(6,*)'i,kx(i),ky(i):',kx(i),ky(i)
cc      end do

cc      write(6,*)'right diaphragm edge angle and gradient'
        inc=3
        do i=1,dia_r_No
          call gradient_east_south_sub(image,nxw,nyh,kx(i),
     %          ky(i),inc,inc,grad,ang)
          gradR(i)=grad
          angR(i)=ang
cc        write(6,*)'grad,ang:',gradR(i),angR(i)
        end do

        grad_ave=0.0
        ang_ave=0.0
        do i=1,dia_r_No
          grad_ave=grad_ave+gradR(i)
          ang_ave=ang_ave+angR(i)
        end do
        grad_ave=grad_ave/float(dia_r_No)
        ang_ave=ang_ave/float(dia_r_No)
cc      write(6,*)'grad_ave,ang_ave',grad_ave,ang_ave

        sig_G=0.0
        sig_A=0.0
        do i=1,dia_r_No
          sig_G=(gradR(i)-grad_ave)*(gradR(i)-grad_ave)+sig_G
          sig_A=(angR(i)-ang_ave)*(angR(i)-ang_ave)+sig_A
        end do
        sig_G=sig_G/float(dia_r_No-1)
        sig_A=sig_A/float(dia_r_No-1)
        sig_G=sqrt(sig_G)
        sig_A=sqrt(sig_A)
cc      write(6,*)'sigma of grad and ang distru. of R. dia'
cc      write(6,*)'sig_G R & sig_A R',sig_G,sig_A

        ideg=3
        ideg1=ideg+1
        ind=0

        do i=1,dia_r_No
          tempx(i)=float(kx(i))
          tempy(i)=float(ky(i))
        end do
        call kofitc(tempx,tempy,dia_r_No,ideg1,cf_r,ideg,ind)

        Rdiaph_No=0
        iiis=feature(14)
        iiie=feature(15)
        do iiii=iiis,iiie
          Rdiaph_No=Rdiaph_No+1
          iix=iiii
          Rdiaph(1,Rdiaph_No)=iix
          call polyfitc_integer(iix,cf_r,ideg,iiy)
          Rdiaph(2,Rdiaph_No)=iiy
        end do
! -------- obtain the right diaphragm edges ------------

ccccccc ******* %%%%%%% &&&&&&& study left lung &&&&&&& %%%%%%% ******* cccccc

c ***  find start point for left diaphragm ***

        index=2
        call straight_line_sub(feature(10),feature(3),feature(12),
```

```
     &       feature(5),xcoord_vert,ycoord_vert,nyh,index)

        lnd1=1
        lnd2=2
        call cross_p_2_line_sub(xcoord_hori,ycoord_hori,nxw,lnd1,
     &       xcoord_vert,ycoord_vert,nyh,lnd2,ixs_D_l,iys_D_l)

c *** end of this part ****

        ix0=ixs_D_l
        iy0=dia_r(2,1)

        xf=ix0
        yf=iy0
        bx=10

        byup=35  !old; upper limit:y=x+12.25 (mm)
        bylow=130!old; lower limit:y=x-45.5  (mm)

c       byup=73  !new; upper limit:y=x+25.7 (mm);(ave+sg)
c       bylow=129!new; lower limit:y=x-45.0 (mm);(ave+2*sg)

! ***   1 pixel = 0.35 mm ****

        count1=1

30      continue

          ix1=xf-bx
          ix2=xf+bx
          if (ix2.ge.nxw) ix2=nxw-1
          iy1=yf-byup
          iy2=yf+bylow
          if (iy2.ge.nyh) iy2=nyh-1

        call Diap_left_ST(image,nxw,nyh,xf,yf,bx,byup,bylow,
     %  ixs1,iys1,ixs2,iys2,count1,ind1,ind2,iRHPV_1stpeak)

cc      write(6,*)'**************************'
cc      write(6,*)'ixs1,iys1,ixs2,iys2',ixs1,iys1,ixs2,iys2
cc      write(6,*)'ind0,ind1,ind2',count1,ind1,ind2
cc      write(6,*)'^^^^^^^^^^^^^^^^^^^^^^^^^^'

        if (count1.ge.2) go to 40

        if (ind2.eq.1) then ! category D, shift ROI half up
          xf=ix0
          iy1=iy0-byup
          iy2=iy1-(byup+bylow)
          yf=iy2+int(float(byup+bylow)/2.0)
          count1=count1+1
          go to 30
        end if

        if (ind2.eq.2) then ! category L, shift ROI half down
          xf=ix0
          yf=iy0+bylow          ! new change in this version 7
          count1=count1+1
          go to 30
        end if

40      continue
```

```
        if (ind2.eq.3) then
          if (ind1.eq.0) then
            itime=1
           else
            itime=2
          end if
        end if

        if (ind1.eq.0) then
          dia_l1(1,1)=ixs1
          dia_l1(2,1)=iys1
        end if

        if (ind1.eq.1) then
          dia_l1(1,1)=ixs1
          dia_l1(2,1)=iys1

          dia_l2(1,1)=ixs2
          dia_l2(2,1)=iys2
        end if
c%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        do j=1, itime
          count=1

c ****** tracing Left dia. outside edges ************
          indd1=1
          if (j.eq.1) then
            xf=ixs1+30
            yf=iys1
           else
            xf=ixs2+30
            yf=iys2
          end if

          byup=40
          bylow=60

          if (xf.gt.feature(17)) go to 60

50        continue

          ix1=xf-bx
          ix2=xf+bx
          if (ix2.ge.nxw) ix2=nxw-1
          iy1=yf-byup
          iy2=yf+bylow
          if (iy2.ge.nyh) iy2=nyh-1

          cond=float(feature(17)-xf)/30.
          if (cond.lt.2.0) then ! check last two point levels
            ind=1
           else
            ind=0
          end if
          call Diaphragm_left(image,nxw,nyh,xf,yf,xc,yc,
     &                        bx,byup,bylow,ind,indd1)

          count=count+1
          if (j.eq.1) then
            dia_l1(1,count)=xc
            dia_l1(2,count)=yc
```

```
           else
            dia_l2(1,count)=xc
            dia_l2(2,count)=yc
          end if
          xf=xc+30 !$$$$ trace L dia. outside edges $$$$
          if (xf.lt.feature(17)) then
            yf=yc
            go to 50
          end if

c ***** tracing Left dia inside edges *******

60        continue
          ind=0
          indd1=2
          if (j.eq.1) then
            xf=dia_l1(1,1)-30
            yf=dia_l1(2,1)
           else
            xf=dia_l2(1,1)-30
            yf=dia_l2(2,1)
          end if
          byup=30
          bylow=30

70        continue

          ix1=xf-bx
          ix2=xf+bx
          if (ix2.ge.nxw) ix2=nxw-1
          iy1=yf-byup
          iy2=yf+bylow
          if (iy2.ge.nyh) iy2=nyh-1

          index=0
          call Diaphragm_left(image,nxw,nyh,xf,yf,xc,yc,
     &                        bx,byup,bylow,index,indd1)

          count=count+1
          if (j.eq.1) then
            dia_l1(1,count)=xc
            dia_l1(2,count)=yc
           else
            dia_l2(1,count)=xc
            dia_l2(2,count)=yc
          end if
          xf=xc-30  !$$$$$ tracing Left dia. inside edges $$$$$$$
          if (xf.ge.feature(16)) then
            yf=yc
            cond=float(xf-feature(16))/30.      ! new add in v.7
            if (cond.lt.3.0) then   ! check last four points y levels
              ind=1
             else
              ind=0
            end if                            ! new add in v. 7
            go to 70
          end if

          if (j.eq.1) then
            dia_l1_No=count
           else
            dia_l2_No=count
          end if
```

```
        end do

c *** end of the part to find left dia. edges following two ST candidates ***

c ********* fit dia_l1 ***********************

cc      write(6,*)'********* fit dia_l1 *************'
cc      write(6,*)'********************* *************'

cc      write(6,*)'rearranged L. dia_l1 edges'
        do i=1,dia_l1_No
          kx(i)=dia_l1(1,i)
          ky(i)=dia_l1(2,i)
        end do
        kxy=100
        call REarrange_S_L(kx,ky,kxy,dia_l1_No)
cc      do i=1,dia_l1_No
cc        write(6,*)'i,kx(i),ky(i):',kx(i),ky(i)
cc      end do

cc      write(6,*)'Left diaphragm edge angle and gradient (L1)'
         inc=3
        do i=1,dia_l1_No
          call gradient_east_south_sub(image,nxw,nyh,kx(i),
     %          ky(i),inc,inc,grad,ang)
          gradL(i)=grad
          angL(i)=ang
cc        write(6,*)'grad,ang:',gradL(i),angL(i)
        end do

        grad_ave=0.0
        ang_ave=0.0
        do i=1, dia_l1_No
          grad_ave=grad_ave+gradL(i)
          ang_ave=ang_ave+angL(i)
        end do
        grad_ave=grad_ave/float(dia_l1_No)
        ang_ave=ang_ave/float(dia_l1_No)
cc      write(6,*)'grad_ave,ang_ave',grad_ave,ang_ave

        sig_G=0.0
        sig_A=0.0
        do i=1,dia_l1_No
          sig_G=(gradL(i)-grad_ave)*(gradL(i)-grad_ave)+sig_G
          sig_A=(angL(i)-ang_ave)*(angL(i)-ang_ave)+sig_A
        end do
        sig_G=sig_G/float(dia_l1_No-1)
        sig_A=sig_A/float(dia_l1_No-1)
        sig_G=sqrt(sig_G)
        sig_A=sqrt(sig_A)
cc      write(6,*)'sigma of grad and ang distru. of Left dia_l1'
cc      write(6,*)'sig_G L. & sig_A L.',sig_G,sig_A

        ideg=3
        ideg1=ideg+1
        ind=0

        do i=1,dia_l1_No
          tempx(i)=float(kx(i))
          tempy(i)=float(ky(i))
        end do
        call kofitc(tempx,tempy,dia_l1_No,ideg1,cf_l,ideg,ind)
```

```
        sig=0.0
        do i=1,dia_l1_No
          call polyfitc_integer(kx(i),cf_l,ideg,iiy)
          sig=(float(iiy-ky(i)))*(float(iiy-ky(i)))+sig
        end do
        sig=sig/(dia_l1_No-1)
        sig=sqrt(sig)
cc      write(6,*)'residual ^ edge points & fitted ones',sig
        sig_L1=sig   !!!!!!!!!

        ideg=1
        ideg1=ideg+1
        ind=0
        call kofitc(tempx,tempy,dia_l1_No,ideg1,cf_l,ideg,ind)
cc      write(6,*)'slope of L1. diaphragm edges:',cf_l(2)
        slope_L1=abs(cf_l(2))   !!!!!!!!

cc      write(6,*)'***** end of fitting of dia_l1 ************'
cc      write(6,*)'********************************************'

c ******** end of fitting of dia_l1 ***********************


c ********* fit dia_l2 ***********************

        if (ind1.eq.1) then

cc      write(6,*)'********* fit dia_l2 *************'
cc      write(6,*)'********************* *************'

cc      write(6,*)'rearranged L. dia_l2 edges'
        do i=1,dia_l2_No
          kx2(i)=dia_l2(1,i)
          ky2(i)=dia_l2(2,i)
        end do
        kxy=100
        call REarrange_S_L(kx2,ky2,kxy,dia_l2_No)
cc      do i=1,dia_l2_No
cc        write(6,*)'i,kx2(i),ky2(i):',kx2(i),ky2(i)
cc      end do

cc      write(6,*)'Left diaphragm edge angle and gradient (L2)'
        inc=3
        do i=1,dia_l2_No
          call gradient_east_south_sub(image,nxw,nyh,kx2(i),
     %          ky2(i),inc,inc,grad,ang)
          gradL(i)=grad
          angL(i)=ang
cc        write(6,*)'grad,ang:',gradL(i),angL(i)
        end do

        grad_ave=0.0
        ang_ave=0.0
        do i=1, dia_l2_No
          grad_ave=grad_ave+gradL(i)
          ang_ave=ang_ave+angL(i)
        end do
        grad_ave=grad_ave/float(dia_l2_No)
        ang_ave=ang_ave/float(dia_l2_No)
cc      write(6,*)'grad_ave,ang_ave',grad_ave,ang_ave

        sig_G=0.0
```

```
        sig_A=0.0
        do i=1,dia_l2_No
          sig_G=(gradL(i)-grad_ave)*(gradL(i)-grad_ave)+sig_G
          sig_A=(angL(i)-ang_ave)*(angL(i)-ang_ave)+sig_A
        end do
        sig_G=sig_G/float(dia_l2_No-1)
        sig_A=sig_A/float(dia_l2_No-1)
        sig_G=sqrt(sig_G)
        sig_A=sqrt(sig_A)
cc      write(6,*)'sigma of grad and ang distru. of Left dia_l2'
cc      write(6,*)'sig_G L. & sig_A L.',sig_G,sig_A

        ideg=3
        ideg1=ideg+1
        ind=0

        do i=1,dia_l2_No
          tempx(i)=float(kx2(i))
          tempy(i)=float(ky2(i))
        end do
        call kofitc(tempx,tempy,dia_l2_No,ideg1,cf_l,ideg,ind)

        sig=0.0
        do i=1,dia_l2_No
          call polyfitc_integer(kx2(i),cf_l,ideg,iiy)
          sig=(float(iiy-ky2(i)))*(float(iiy-ky2(i)))+sig
        end do
        sig=sig/(dia_l2_No-1)
        sig=sqrt(sig)
cc      write(6,*)'residual ^ edge points & fitted ones',sig
        sig_L2=sig

        ideg=1
        ideg1=ideg+1
        ind=0
        call kofitc(tempx,tempy,dia_l2_No,ideg1,cf_l,ideg,ind)
cc      write(6,*)'slope of L2 diaphragm edges:',cf_l(2)
        slope_L2=abs(cf_l(2))

cc      write(6,*)'***** end of fitting of dia_l2 ************'
cc      write(6,*)'********************************************'

        end if

c ******** end of fitting of dia_l2 ***********************

c *** add rule-based method ******************

        index=0

        if (ind1.eq.0) then !!!!!! only one SP in left
cc          write(6,*) '***L1 is good***'
            index=1
            go to 1000
        end if


        if (ind1.eq.1) then !!!!! two SP in left
          if ((slope_L1.le.0.35).and.(slope_L2.gt.0.35)) then
cc          write(6,*) '***L1 is good***'
            index=1
            go to 1000
          end if
```

```
        end if

        if (ind1.eq.1) then !!!! two SP in left
          if ((slope_L2.le.0.35).and.(slope_L1.gt.0.35)) then
cc          write(6,*) '***L2 is good***'
            index=2
            go to 1000
          end if
        end if

        if (ind1.eq.1) then  !!!!! two SP in left
        if (((slope_L1.ge.0.35).and.(slope_l2.ge.0.35)).
     %   or.((slope_L1.le.0.35).and.(slope_l2.le.0.35))) then

           if (sig_l1.lt.sig_l2) then
cc          write(6,*) '***L1 is good***'
            index=1
           else if (sig_l2.lt.sig_l1) then
cc          write(6,*) '***L2 is good***'
            index=2
           else
cc          write(6,*) '***L1 is good***'
            index=1
           end if

          end if
        end if

1000    continue

cc      write(6,*)'index(left dia selection):',index

        if (index.eq.1) then
          do i=1,dia_l1_No
            tempx(i)=float(kx(i))
            tempy(i)=float(ky(i))
          end do
          ideg=3
          ideg1=ideg+1
          ind=0
          call kofitc(tempx,tempy,dia_l1_No,ideg1,cf_l,ideg,ind)

          Ldiaph_No=0
          iiis=feature(16)
          iiie=feature(17)
          do iiii=iiis,iiie
            Ldiaph_No=Ldiaph_No+1
            iix=iiii
            Ldiaph(1,Ldiaph_No)=iix
            call polyfitc_integer(iix,cf_l,ideg,iiy)
            Ldiaph(2,Ldiaph_No)=iiy
          end do

         else if (index.eq.2) then
          do i=1,dia_l2_No
            tempx(i)=float(kx2(i))
            tempy(i)=float(ky2(i))
          end do
          ideg=3
          ideg1=ideg+1
          ind=0
          call kofitc(tempx,tempy,dia_l1_No,ideg1,cf_l,ideg,ind)
          Ldiaph_No=0
          iiis=feature(16)
          iiie=feature(17)
          do iiii=iiis,iiie
            Ldiaph_No=Ldiaph_No+1
            iix=iiii
            Ldiaph(1,Ldiaph_No)=iix
            call polyfitc_integer(iix,cf_1,ideg,iiy)
            Ldiaph(2,Ldiaph_No)=iiy
          end do

         else
          write(6,*)'the rule-based selection is fail'
        end if

        return
        end
c***************************************************************************
c
c       Diaphragm_Right.for
c
c***************************************************************************


        subroutine Diaphragm_Right(image,nxw,nyh,xf,yf,xc,yc,
     &                            bx,byup,bylow,ind1,ind2)

        implicit integer*2 (i-n)

        integer*2 image(nxw,nyh),   ! image buffer
     &            xf,yf,            ! former dia. edge point x & y coordinates
     &            xc,yc,            ! current dia. edge point x&y coordinates
     &            bx,byup,bylow,    ! half size of search boxs
     &            ind1,             ! compare yc and yf; =0 no comparison;
                                    ! =1, do comparison;if yc le. yf, corr. yc
     &            ind2              !=1, trac outside; =2 trac inside;
c***************************************************************************

        real prof(500),prof_smo(500),fd(500)
        integer*2 posi_fd(500),pp_min,pp_max

        real*8 sum

        do i=1,500
           prof(i)=0.0
           prof_smo(i)=0.0
           fd(i)=0.0
           posi_fd(i)=0
        end do

        xc=xf

        ix_start=xf-bx
          if (ix_start.le.1) ix_start=2
        ix_end=xf+bx
        iy_start=yf-byup
        iy_end=yf+bylow
          if (iy_end.ge.nyh) iy_end=nyh-1

c **** range for profle calculation ****

        sum=0.0
```

```
        do iy=iy_start,iy_end
          k=iy-iy_start+1
          do ix=ix_start,ix_end
            sum=sum+float(image(ix,iy))
          end do
          sum=sum/float(ix_end-ix_start+1)
          prof(k)=sum
          sum=0.0
        end do

        NN=iy_end-iy_start+1
        mm=500
        inc=15
        call prof_smo_sub(prof,mm,NN,inc,prof_smo)

        inc2=3
        call fd_north_sub(prof_smo,posi_fd,fd,mm,nn,inc2,No_fd)

c *** &&& find right diaphragm edge points &&& ***

        fd_min=fd(1)
        pp_min=1
        do k=2,No_fd
          if (fd(k).lt.fd_min) then
            fd_min=fd(k)
            pp_min=k
          end if
        end do

        yc=iy_start+posi_fd(pp_min)-1

        if (ind1.eq.1) then
          if (yc.le.yf) then
            do k=pp_min+1, No_fd-1
              k1=k+1
              if (fd(k1).lt.fd(k)) then
                k_search=k1
                go to 5
              end if
            end do
            if (ind2.eq.1) then
              yc=yf+17        ! from data base
             else
              yc=yf
            end if
            go to 10

5           continue

            fd_min=fd(k_search)
            pp_min=k_search
            do k=k_search+1,No_fd
              if (fd(k).lt.fd_min) then
                fd_min=fd(k)
                pp_min=k
              end if
            end do
            yc=iy_start+posi_fd(pp_min)-1
          end if
        end if


10      continue
```

```
        return
        end

c***************************************************************************
c
c       Diaphragm_left.for
c
c***************************************************************************


        subroutine Diaphragm_left(image,nxw,nyh,xf,yf,xc,yc,
     &                            bx,byup,bylow,ind1,ind2)

        implicit integer*2 (i-n)

        integer*2 image(nxw,nyh),   ! image buffer
     &            xf,yf,            ! former dia. edge point x & y coordinates
     &            xc,yc,            ! current dia. edge point x&y coordinates
     &            bx,byup,bylow,    ! half size of search boxs
     &            ind1,             ! compare yc and yf; =0 no comparison;
                                    ! =1, do comparison;if yc le. yf, corr. yc
     %            ind2              !=1, trac outside; =2,trac inside;
c***************************************************************************

        real prof(500),prof_smo(500),fd(500)
        integer*2 posi_fd(500),pp_min,pp_max

        real*8 sum

        do i=1,500
           prof(i)=0.0
           prof_smo(i)=0.0
           fd(i)=0.0
           posi_fd(i)=0
        end do

        xc=xf

        ix_start=xf-bx
          if (ix_start.le.1) ix_start=2
        ix_end=xf+bx
        iy_start=yf-byup
        iy_end=yf+bylow
          if (iy_end.ge.nyh) iy_end=nyh-1

c **** range for profle calculation ****

        sum=0.0
        do iy=iy_start,iy_end
          k=iy-iy_start+1
          do ix=ix_start,ix_end
            sum=sum+float(image(ix,iy))
          end do
          sum=sum/float(ix_end-ix_start+1)
          prof(k)=sum
          sum=0.0
        end do

        NN=iy_end-iy_start+1
        mm=500
        inc=15
```

```
        call prof_smo_sub(prof,mm,NN,inc,prof_smo)

        inc2=3
        call fd_north_sub(prof_smo,posi_fd,fd,mm,nn,inc2,No_fd)

c *** &&& find left diaphragm edge points &&& ***

        fd_min=fd(1)
        pp_min=1
        do k=2,No_fd
          if (fd(k).lt.fd_min) then
            fd_min=fd(k)
            pp_min=k
          end if
        end do

        yc=iy_start+posi_fd(pp_min)-1

        if (ind1.eq.1) then
          if (yc.le.yf) then
            do k=pp_min+1, No_fd-1
              k1=k+1
              if (fd(k1).lt.fd(k)) then
                k_search=k1
                go to 5
              end if
            end do
            if (ind2.eq.1) then
              yc=yf+17        ! from data base
             else
              yc=yf
            end if
            go to 10

5           continue

            fd_min=fd(k_search)
            pp_min=k_search
            do k=k_search+1,No_fd
              if (fd(k).lt.fd_min) then
                fd_min=fd(k)
                pp_min=k
              end if
            end do
            yc=iy_start+posi_fd(pp_min)-1
          end if
        end if

10      continue

        return
        end
c***************************************************************************
c
c       histogram_analysis.for
c
c***************************************************************************
c
c
        subroutine histogram_analysis(hist_pixel,hist_freq,No,bin,
     %                              PV_1stpeak)
```

```
      implicit integer*2 (i-n)

      integer*2 hist_pixel(1024),
     &          bin,
     &          No,
     %          PV_1stpeak    ! output; the PV at 1st peak position of the
                              ! histogram

      real      hist_freq(1024)
c*****************************************************************************

      integer*2 posi_fd_hist(1024),p_low,p_up,p_med,width

      real      fd_hist(1024)

      do i=1,1024
         posi_fd_hist(i)=0
         fd_hist(i)=0.0
      end do

c *** width of histgram: from 1% accumalation of both sides ***

      accum=0.0
      do i=1,No
        accum=accum+hist_freq(i)*float(bin)
        if (accum.ge.1.0) then
          p_low=hist_pixel(i)
          go to 5
        end if
      end do
5     continue

      accum=0.0
      do i=No,1,-1
        accum=accum+hist_freq(i)*float(bin)
        if (accum.ge.1.0) then
          p_up=hist_pixel(i)
          go to 10
        end if
      end do
10    continue

      width=p_up-p_low
cc    write(6,*)'width:',width,p_low,p_up

c *** end of this part ***

c *** find Number of peak in the histogram ****

      N_hist=1024
      lnd=1
      call fd_south_sub(hist_freq,posi_fd_hist,fd_hist,
     &                  N_hist,No,lnd,No_fd_hist)


      No_peak=0
      do i=1,No_fd_hist-1
        i1=i+1
        if ((fd_hist(i).ge.0.0).and.(fd_hist(i1).lt.0.0)) then
          No_peak=No_peak+1
        end if
```

```
      end do
cc    write(6,*)'No. of peak:',No_peak

      No_valley=-1
      do i=1,No_fd_hist-1
        i1=i+1
        if ((fd_hist(i).le.0.0).and.(fd_hist(i1).gt.0.0)) then
          No_valley=No_valley+1
        end if
      end do
cc    write(6,*)'No. of valley:',No_valley

c *** end of this part ***

c *** find max peak position, i.e. the pixel value at
c     the position where has max histogram value. ***

      max_pixel=hist_pixel(1)
      freq_max=hist_freq(1)
      do i=2,No
        if (hist_freq(i).gt.freq_max) then
          freq_max=hist_freq(i)
          max_pixel=hist_pixel(i)
        end if
      end do
cc    write(6,*)'max_pixel',max_pixel !PV where has max. hist. value

c *** end of this part ***

c *** find maximum pixel at peak, i.e. the pixel value at the
c     1st peak position of the histogram ***

      do i=No_fd_hist,2,-1
        i1=i-1
        if ((fd_hist(i).lt.0.0).and.(fd_hist(i1).gt.0.0)) then
          max_pixel_peak=posi_fd_hist(i1)*bin
          go to 20
        end if
      end do
20    continue
cc    write(6,*)'max_pixel_peak',max_pixel_peak
      PV_1stpeak=max_pixel_peak ! PV where is 1st peak posi.

c *** find minimum pixel at peak ***

      do i=1,No_fd_hist-1
        i1=i+1
        if ((fd_hist(i).gt.0.0).and.(fd_hist(i1).lt.0.0)) then
          min_pixel_peak=posi_fd_hist(i)*bin
          go to 30
        end if
      end do
30    continue
cc    write(6,*)'min_pixel_peak',min_pixel_peak | PV where is 2nd peak

c *** end of this part ***

c *** part of find mediam pixel value of histogram ***

      accum=0.0
      do i=1,No
        accum=accum+hist_freq(i)*float(bin)
        if (accum.ge.50.0) then
```

```
          p_med=hist_pixel(i)
          go to 40
        end if
      end do
40    continue
cc    write(6,*)'mediam pixel value of hist:',p_med

      return
      end

c*****************************************************************************
c
c     Diap_Left_ST.for (a subroutine for finding starting point of
c                       left diaphragm edges)
c
c*****************************************************************************


      subroutine Diap_Left_ST(image,nxw,nyh,xf,yf,bx,byup,bylow,
     &       ixs1,iys1,ixs2,iys2,ind0,ind1,ind2,iRHPV_1stPeak)

      implicit integer*2 (i-n)

      integer*2 image(nxw,nyh),   ! image buffer
     &          xf,yf,            ! initial input of x,y cood for L dia.
     &          ixs1,iys1,        ! 1st L dia. ST.
     %          ixs2,iys2,        ! 2nd L dia. ST.
     &          bx,byup,bylow,    ! half size of search boxs
     %          ind0,             ! =1,1st time call;=2 2nd time due to shift
     &          ind1,             ! =0,edge ratio R<= 0.55; one ST. (1st)
                                  ! =1,edge ratio R> 0.55; two STs
     %          ind2,             ! =1,cate D; =2 cate L; =3 other cate.
     %          iRHPV_1stPeak     ! input; The PV at 1st peak posi of right
                                  !diap. ST roi,for cate. L and D classifi.
c*****************************************************************************

      real prof(500),prof_smo(500),fd(500),fd_smo(500)
      integer*2 posi_fd(500)

      real hist_freq(1024),hist_freq_smo(1024)
      integer*2 hist_pixel(1024)

      real*8 sum

c*****************************************************************************

      do i=1,500
         prof(i)=0.0
         prof_smo(i)=0.0
         fd(i)=0.0
         fd_smo(i)=0.0
         posi_fd(i)=0
      end do

      ind2=3

      ixs1=xf
      ixs2=xf

      ix_start=xf-bx
        if (ix_start.le.1) ix_start=2
      ix_end=xf+bx
      iy_start=yf-byup
```

```
      iy_end=yf+bylow
        if (iy_end.ge.nyh) iy_end=nyh-1

c ***** above is search ROI *************

c ***** histogram in the ROI ************

      ix1=ix_start
      ix2=ix_end
      iy1=iy_start
      iy2=iy_end
      incre=25
      call histogram_roi_sub(image,nxw,nyh,ix1,ix2,iy1,iy2,
     &     incre,No,hist_pixel,hist_freq)

cc    write(6,*)'*** left lung histogram info. at start point ***'
       N_h=1024
       inc=3
      call prof_smo_sub(hist_freq,N_h,No,inc,hist_freq_smo)

      call histogram_analysis(hist_pixel,hist_freq_smo,No,
     %                  incre,iLHPV_1stpeak)

c **** finish of this part: analysis of the Histogram in ROI *******

c **** range for profle calculation ****

      sum=0.0
      do iy=iy_start,iy_end
        k=iy-iy_start+1
        do ix=ix_start,ix_end
          sum=sum+float(image(ix,iy))
        end do
        sum=sum/float(ix_end-ix_start+1)
        prof(k)=sum
        sum=0.0
      end do

      NN=iy_end-iy_start+1
      kk=15
      mm=500
      inc=3
      call prof_smo_sub(prof,mm,NN,kk,prof_smo)

      call fd_north_sub(prof_smo,posi_fd,fd,mm,nn,inc,No_fd)
      m=int(float(inc-1)/2.0)
      call prof_smo_sub(fd,mm,NN,inc,fd_smo)

c *** &&& find left diaphragm edge starting points &&& ***


c *** find first min: min_F,fd_min_F ***

      fd_min_F=fd_smo(m+1)
      min_F=m+1
      do k=m+2,No_fd-m
        if (fd_smo(k).lt.fd_min_F) then
          fd_min_F=fd_smo(k)
          min_F=k
        end if
      end do

cc    write(6,*)'P. min_F;fd_min_F',posi_fd(min_F),fd_min_F
```

```
        ! min_F is related to iys1 ********************
c *** end of this part ***


c *** find range of firts min ***

        do j=min_F,m+2,-3
          j1=j-3
          if (fd_smo(j1).lt.fd_smo(j)) then
            limit_left=j1
            go to 5
          end if
        end do
        limit_left=0
cc      write(6,*)'left side no change'

5       continue

        do j=min_F,No_fd-(m+1),3
          j1=j+3
          if (fd_smo(j1).lt.fd_smo(j)) then
            limit_right=j1
            go to 10
          end if
        end do
        limit_right=0
cc      write(6,*)'right side no cheange'

10      continue

cc      if ((limit_left.ne.0).and.(limit_right.ne.0)) then
cc      write(6,*)'range of first min:',posi_fd(limit_left),
cc     %posi_fd(limit_right)
cc      end if

c *** end of this part ***

c *** find second min position: min_S, fd_min_S ***

        if (limit_left.ne.0) then
        min_left=m+1
        fd_min_left=fd_smo(m+1)
        do k=m+2,limit_left
          if (fd_smo(k).lt.fd_min_left) then
            fd_min_left=fd_smo(k)
            min_left=k
          end if
        end do
cc      write(6,*)'fd_min_left,min_left',fd_min_left,posi_fd(min_left)
        end if

        if (limit_right.ne.0) then
        min_right=limit_right
        fd_min_right=fd_smo(limit_right)
        do k=limit_right+1, No_fd-m
          if (fd_smo(k).lt.fd_min_right) then
            fd_min_right=fd_smo(k)
            min_right=k
          end if
        end do
cc      write(6,*)'fd_min_right,min_right',fd_min_right,posi_fd(min_right)
        end if
```

```
        if ((limit_left.ne.0).and.(limit_right.ne.0)) then
        if (fd_min_left.le.fd_min_right) then
          fd_min_S=fd_min_left
          min_S=min_left
         else
          fd_min_S=fd_min_right
          min_S=min_right
        end if
        end if

        if ((limit_left.ne.0).and.(limit_right.eq.0)) then
          min_S=min_left
          fd_min_S=fd_min_left
        end if

        if ((limit_left.eq.0).and.(limit_right.ne.0)) then
          min_S=min_right
          fd_min_S=fd_min_right
        end if

        if ((limit_left.eq.0).and.(limit_right.eq.0)) then
          min_S=min_F
          fd_min_S=0.0
        end if

cc      write(6,*)'P. min_S;fd_min_S',posi_fd(min_S),fd_min_S
        !       min_S=posi_fd(min_S) is related iys2.

        ratio=fd_min_S/fd_min_F
        Idist=posi_fd(min_S)-posi_fd(min_F)

cc      write(6,*)'ratio=fd_min_S/fd_min_F:',ratio
cc      write(6,*)'distance=min_S-min_F:',Idist

c ***************** end of this part ***************

c **** max FD value between min_S and min_F ****

        if (min_S.lt.min_F) then
          istart=min_S
          iend=min_F
         else
          istart=min_F
          iend=min_S
        end if
        fd_max=fd_smo(istart)
        max_fd=istart
        do i=istart+1,iend
          if (fd_smo(i).gt.fd_max) then
            fd_max=fd_smo(i)
            max_fd=i
          end if
        end do
cc      write(6,*)'P. max_FD;max_fd in RANGE:',posi_fd(max_fd),fd_max

c ********************************************************

        If (ratio.le.0.55) then
           iys1=iy_start+posi_fd(min_F)-1
           ind1=0
           go to 110
          else
```

```
           if (ind0.ge.2) then
             iys1=iy_start+posi_fd(min_F)-1
             iys2=iy_start+posi_fd(min_S)-1
             ind1=1
             ind2=3
             go to 110
           end if
        end if

C**************** appling rule-based scheme only in 1st call ***************

        abs_1stmin=abs(fd_min_F)
        abs_2ndmin=abs(fd_min_S)
        if ((abs_2ndmin.le.1.72).and.(abs_1stmin.le.2.61)) then
          if (fd_max.gt.6.0) then
cc          write(6,*)' category of LG'
            ind1=1
            go to 20
          end if
cc        write(6,*)'could be cate. L and D'
          thresh=1.444*float(iRHPV_1stpeak)-465.444
          iLHth=int(thresh)
          if (iLHPV_1stpeak.ge.iLHth) then
            ind1=1
            ind2=1
cc          write(6,*)'category D'
            go to 110
          end if
          if (iLHPV_1stpeak.lt.iLHth) then
            ind1=1
            ind2=2
cc          write(6,*)'category L'
            go to 110
          end if
        end if

cc      write(6,*)'categories:A2,NG,C'
20      continue

        iys1=iy_start+posi_fd(min_F)-1
        iys2=iy_start+posi_fd(min_S)-1
        ind1=1
        ind2=3

c *** end of this part ***


110     continue

        return
        end
```

```
/*===================================================================
Name            DicomUtil.c
Function        Utility for Reading Dicom Header
Author          SHIGE   04/27/98
                Shige   Modified        06/17/99

The following functions are included,
void            getKanaTable
unsigned short  getShortInt
unsigned long   getLongInt
void            putShortInt
void            putLongInt
void            putCharStr
void            elimSpace
void            ShortCutTime
void            ModifyPatName
struct FcrDicom getFcrDicomInfo
struct ThvDicom getThvDicomInfo
void            putFcrDicomInfo
void            putThvDicomInfo
===================================================================*/
#include        <stdio.h>
#include        <string.h>
#include        <stdlib.h>
#include        <ctype.h>
#include        <math.h>
#include        "TempSub.H"

#ifdef BYTE_ORDER                  // BYTE_ORDER is defined by /usr/includes/sys
#if BYTE_ORDER == LITTLE_ENDIAN
 int _isMachineLowEndian = 1;
#else
 int _isMachineLowEndian = 0;
#endif
#else
 int _isMachineLowEndian = 0;
#endif

#define         CLEN    64
#define         NCHAR   101

char            charTab[NCHAR][4];

/*============================================
        getKanaTable.c
        11/03/98        Coded by Shige
============================================*/
void    getKanaTable()
{
strcpy(charTab[  0],"WO");  strcpy(charTab[  1],"A");
strcpy(charTab[  2],"I");   strcpy(charTab[  3],"U");
strcpy(charTab[  4],"E");   strcpy(charTab[  5],"O");
strcpy(charTab[  6],"YA");  strcpy(charTab[  7],"YU");
strcpy(charTab[  8],"YO");  strcpy(charTab[  9],"TSU");
strcpy(charTab[ 10],"X");   strcpy(charTab[ 11],"A");
strcpy(charTab[ 12],"I");   strcpy(charTab[ 13],"U");
strcpy(charTab[ 14],"E");   strcpy(charTab[ 15],"O");
strcpy(charTab[ 16],"KA");  strcpy(charTab[ 17],"KI");
strcpy(charTab[ 18],"KU");  strcpy(charTab[ 19],"KE");
strcpy(charTab[ 20],"KO");  strcpy(charTab[ 21],"SA");
strcpy(charTab[ 22],"SHI"); strcpy(charTab[ 23],"SU");
strcpy(charTab[ 24],"SE");  strcpy(charTab[ 25],"SO");
strcpy(charTab[ 26],"TA");  strcpy(charTab[ 27],"CHI");
```

```
strcpy(charTab[ 28],"TSU"); strcpy(charTab[ 29],"TE");
strcpy(charTab[ 30],"TO");  strcpy(charTab[ 31],"NA");
strcpy(charTab[ 32],"NI");  strcpy(charTab[ 33],"NU");
strcpy(charTab[ 34],"NE");  strcpy(charTab[ 35],"NO");
strcpy(charTab[ 36],"HA");  strcpy(charTab[ 37],"HI");
strcpy(charTab[ 38],"FU");  strcpy(charTab[ 39],"HE");
strcpy(charTab[ 40],"HO");  strcpy(charTab[ 41],"MA");
strcpy(charTab[ 42],"MI");  strcpy(charTab[ 43],"MU");
strcpy(charTab[ 44],"ME");  strcpy(charTab[ 45],"MO");
strcpy(charTab[ 46],"YA");  strcpy(charTab[ 47],"YU");
strcpy(charTab[ 48],"YO");  strcpy(charTab[ 49],"RA");
strcpy(charTab[ 50],"RI");  strcpy(charTab[ 51],"RU");
strcpy(charTab[ 52],"RE");  strcpy(charTab[ 53],"RO");
strcpy(charTab[ 54],"WA");  strcpy(charTab[ 55],"N");
strcpy(charTab[ 56],"X");   strcpy(charTab[ 57],"X");
strcpy(charTab[ 58],"X");   strcpy(charTab[ 59],"X");
strcpy(charTab[ 60],"X");   strcpy(charTab[ 61],"X");
strcpy(charTab[ 62],"X");   strcpy(charTab[ 63],"X");
strcpy(charTab[ 64],"X");   strcpy(charTab[ 65],"X");
strcpy(charTab[ 66],"GA");  strcpy(charTab[ 67],"GI");
strcpy(charTab[ 68],"GU");  strcpy(charTab[ 69],"GE");
strcpy(charTab[ 70],"GO");  strcpy(charTab[ 71],"ZA");
strcpy(charTab[ 72],"JI");  strcpy(charTab[ 73],"ZU");
strcpy(charTab[ 74],"ZE");  strcpy(charTab[ 75],"ZO");
strcpy(charTab[ 76],"DA");  strcpy(charTab[ 77],"DI");
strcpy(charTab[ 78],"DU");  strcpy(charTab[ 79],"DE");
strcpy(charTab[ 80],"DO");  strcpy(charTab[ 81],"X");
strcpy(charTab[ 82],"X");   strcpy(charTab[ 83],"X");
strcpy(charTab[ 84],"X");   strcpy(charTab[ 85],"X");
strcpy(charTab[ 86],"BA");  strcpy(charTab[ 87],"BI");
strcpy(charTab[ 88],"BU");  strcpy(charTab[ 89],"BE");
strcpy(charTab[ 90],"BO");  strcpy(charTab[ 91],"X");
strcpy(charTab[ 92],"X");   strcpy(charTab[ 93],"X");
strcpy(charTab[ 94],"X");   strcpy(charTab[ 95],"X");
strcpy(charTab[ 96],"PA");  strcpy(charTab[ 97],"PI");
strcpy(charTab[ 98],"PU");  strcpy(charTab[ 99],"PE");
strcpy(charTab[100],"PO");
}

/*==================================================================
        Function getShortInt
==================================================================*/
unsigned short getShortInt(char *buf)
{
  /* We are writing DICOM, so pixels should be stored as low-endian.  Thus,
     if the machine is high endian, we need to byte swap. */
  int shouldByteSwap = !_isMachineLowEndian;
char            c;
unsigned short  *us;
  if (shouldByteSwap)
    {
    c  = buf[0]; buf[0] = buf[1]; buf[1] = c;
    }
us = (unsigned short*)buf;
return *us;
}

/*==================================================================
        Function getLongInt
==================================================================*/
unsigned long getLongInt(char *buf)
{
  /* We are writing DICOM, so pixels should be stored as low-endian.  Thus,
```

```
     if the machine is high endian, we need to byte swap. */
  int shouldByteSwap = !_isMachineLowEndian;
char            c;
unsigned long   *ul;
  if (shouldByteSwap)
    {
    c  = buf[0]; buf[0] = buf[3]; buf[3] = c;
    c  = buf[1]; buf[1] = buf[2]; buf[2] = c;
    }
ul = (unsigned long*)buf;
return *ul;
}

/*==================================================================
        Function putShortInt
==================================================================*/
void putShortInt(char *header, unsigned long ph,
                               unsigned long dLen,
                               unsigned short val)
{
  /* We are writing DICOM, so pixels should be stored as low-endian.  Thus,
     if the machine is high endian, we need to byte swap. */
  int shouldByteSwap = !_isMachineLowEndian;
char            *pc;
unsigned long   i;

if (shouldByteSwap)
  {
  val = ((val & 0xFF00) >> 8) | ((val & 0x00FF) << 8);
  }
pc  = (char*)&val;
for (i = 0; i < dLen; i++) header[i + ph] = pc[i];
}

/*==================================================================
        Function putLongInt
==================================================================*/
void putLongInt(char *header, unsigned long ph,
                              unsigned long dLen,
                              unsigned long val)
{
  /* We are writing DICOM, so pixels should be stored as low-endian.  Thus,
     if the machine is high endian, we need to byte swap. */
  int shouldByteSwap = !_isMachineLowEndian;
char            *pc;
unsigned long   i;

if (shouldByteSwap)
  {
  val = ((val & 0xFF000000) >> 24) |
        ((val & 0x00FF0000) >>  8) |
        ((val & 0x0000FF00) <<  8) |
        ((val & 0x000000FF) << 24);
  }
pc  = (char*)&val;
for (i = 0; i < dLen; i++) header[i + ph] = pc[i];
}

/*==================================================================
        Function putCharStr
==================================================================*/
void putCharStr(char *header, unsigned long ph,
                              unsigned long dLen,
```

```
                              char *buf)
{
unsigned long   i;
unsigned long   bLen;

for (i = 0; i < dLen; i++) header[i + ph] = ' ';
bLen = strlen(buf);
if (bLen <= dLen) {
  for (i = 0; i < bLen; i++) header[i + ph] = buf[i];
  }
else {
  for (i = 0; i < dLen; i++) header[i + ph] = buf[i];
  }
}

/*==================================================================
        Function elimSpace
==================================================================*/
void elimSpace(char *buf, unsigned long *n)
{
int     i, c;

for (i = 0; i < *n; i++) {
  if (buf[i] == ' ') buf[i] = 0;
  else               break;
  }
for (i = *n - 1; i >= 0; i--) {
  if (buf[i] == ' ') buf[i] = 0;
  else               break;
  }

c = 0;
for (i = 0; i < *n; i++) {
  if (buf[i] != 0) buf[c++] = buf[i];
  }
buf[c] = 0;
*n = c;
}

/*==================================================================
        Function ShortCutTime(char *TexTime)
==================================================================*/
void ShortCutTime(char *TexTime)
{
char    buf[128];
extern  int trimcha(char*, char*, char, char);

strcpy(buf, TexTime);
(void)trimcha(TexTime, buf, '/', '.');
}

/*==================================================================
        Function ModifyPatName
        Remarks:        Alphanumeric characters
                        are not affected.
==================================================================*/
void ModifyPatName(char *buf, unsigned long *nc)
{
int     i, j, cl, cf;
int     flag;
char    LastName[128], FirstName[128];
char    auf[128];
int     p, n;
```

```
char    k;

/*--------------------------------------------------
        Last Name
--------------------------------------------------*/
cl = flag = 0;
for (i = 0; i < *nc; i++) {
  if (isalnum((int)buf[i]) ||
    ((unsigned char)buf[i] >= 0xA6 && (unsigned char)buf[i] <= 0xDF &&
     (unsigned char)buf[i] != 0xB0)) {
    LastName[cl++] = buf[i];
    flag = 1;
    }
  else if (flag != 0) goto LABEL1;
  }
if (flag == 0) {
  printf("No Patient Name\n");
  strcpy(buf, "UnknownPatientName");
  *nc = 18;
  return;
  }
else {
  LastName[cl] = 0;
  strcpy(auf, LastName);
  *nc = cl;
  goto LABEL3;
  }

/*--------------------------------------------------
        First Name
--------------------------------------------------*/
LABEL1:
LastName[cl] = 0;
cf = flag = 0;
for (j = i + 1; j < *nc; j++) {
  if (isalnum((int)buf[j]) ||
    ((unsigned char)buf[j] >= 0xA6 && (unsigned char)buf[j] <= 0xDF &&
     (unsigned char)buf[j] != 0xB0)) {
    FirstName[cf++] = buf[j];
    flag = 1;
    }
  else if (flag != 0) goto LABEL2;
  }
if (flag == 0) {
  strcpy(auf, LastName);
  *nc = cl;
  goto LABEL3;
  }
LABEL2:
FirstName[cf] = 0;
sprintf(auf, "%s_%s", LastName, FirstName);
*nc = cl + cf + 1;

/*--------------------------------------------------
        Conversion of Kana to Alphabet
--------------------------------------------------*/
LABEL3:
p = 0;
for (j = 0; j < *nc; j++) {
  if ((unsigned char)auf[j] != 0xDE && (unsigned char)auf[j] != 0xDF) {
    if ( !(isalnum((int)auf[j]) || (unsigned char)auf[j] == '_') ) {
      if ((unsigned char)auf[j + 1] != 0xDE &&
          (unsigned char)auf[j + 1] != 0xDF) {
```

```
        k = (unsigned char)auf[j] - (unsigned char)0xA6;
        n = strlen(charTab[k]);
        for (i = 0; i < n; i++) buf[p++] = charTab[k][i];
        }
      else if ((unsigned char)auf[j + 1] == 0xDE) {
        k = (unsigned char)auf[j] - (unsigned char)0xA6 + 50;
        n = strlen(charTab[k]);
        for (i = 0; i < n; i++) buf[p++] = charTab[k][i];
        }
      else if ((unsigned char)auf[j + 1] == 0xDF) {
        k = (unsigned char)auf[j] - (unsigned char)0xA6 + 60;
        n = strlen(charTab[k]);
        for (i = 0; i < n; i++) buf[p++] = charTab[k][i];
        }
      }
    else buf[p++] = auf[j];
    }
  }
buf[p] = 0;
*nc = p;
}

/*=================================================================
        Function getFcrDicomInfo()
        Remarks:        Kana characters and alphabets are
                        available in the Patient's Name.
=================================================================*/
struct FcrDicom getFcrDicomInfo(FILE *fp, char *FcrDcmHeader)
{
int     i;
int     ns, nl;
char    *buf;
FILE    *fq;

unsigned short  gTag, eTag;
unsigned long   dLen, gLen;

unsigned short  getShortInt(char*);
unsigned long   getLongInt(char*);
void            elimSpace(char*, unsigned long*);
void            ModifyPatName(char*, unsigned long*);
void            getKanaTable();
void            ShortCutTime(char*);

char    PatBDate[CLEN];         /* Patient Birth Date */
char    PatSex[CLEN];           /* Patient Sex */
char    EDRMode[CLEN];          /* EDR Mode */
int     ImageSize;              /* Image  Size (Bytes) */
int     HeaderSize;             /* Header Size (Bytes) */

struct FcrDicom fcr;

/*-------------------------------------------------------------
        Read Kana Table
-------------------------------------------------------------*/
ns = 2; nl = 4; HeaderSize = 0;
fcr.Date[0] = fcr.Time[0] = fcr.PatName[0] = fcr.PatID[0] = 0;
buf = new char[MAX_FCR_DCM_HEADER_SIZE];

getKanaTable();

/*-------------------------------------------------------------
        Read DICOM Header
```

```
-------------------------------------------------------------*/
while (ns == (int)fread(buf, sizeof(char), ns, fp)) {
  for (i = 0; i < ns; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
  HeaderSize = HeaderSize + ns;
  gTag = getShortInt(buf);
  fread(buf, sizeof(char), ns, fp);
  for (i = 0; i < ns; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
  HeaderSize = HeaderSize + ns;
  eTag = getShortInt(buf);
  fread(buf, sizeof(char), nl, fp);
  for (i = 0; i < nl; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
  HeaderSize = HeaderSize + nl;
  dLen = getLongInt(buf);

  if    (gTag == STUDY_GROUP) {
    switch (eTag) {
      case      DATE_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) fcr.Date[i] = buf[i];
        fcr.Date[dLen] = 0;
        elimSpace(fcr.Date, &dLen);
        break;
      case      TIME_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) fcr.Time[i] = buf[i];
        fcr.Time[dLen] = 0;
        elimSpace(fcr.Time, &dLen);
        ShortCutTime(fcr.Time);
        break;
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else if (gTag == PATIENT_GROUP) {
    switch (eTag) {
      case      NAME_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) fcr.PatName[i] = buf[i];
        fcr.PatName[dLen] = 0;
        ModifyPatName(fcr.PatName, &dLen);
        break;
      case      ID_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) fcr.PatID[i] = buf[i];
        fcr.PatID[dLen] = 0;
        elimSpace(fcr.PatID, &dLen);
        break;
      case      BIRTH_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) PatBDate[i] = buf[i];
        PatBDate[dLen] = 0;
```

```
        break;
      case      SEX_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) PatSex[i] = buf[i];
        PatSex[dLen] = 0;
        break;
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else if (gTag == EDR_GROUP) {
    switch (eTag) {
      case      MODE_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) EDRMode[i] = buf[i];
        EDRMode[dLen] = 0;
        break;
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else if (gTag == IMAGE_GROUP) {
    switch (eTag) {
      case      ROWS_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        fcr.Row = (int)getShortInt(buf);
        break;
      case      COLUMNS_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        fcr.Col = (int)getShortInt(buf);
        break;
      case      DEPTH_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        fcr.NBit = (int)getShortInt(buf);
        break;
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else if (gTag == IP_GROUP) {
    switch (eTag) {
      case      SCAN_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) fcr.Dir[i] = buf[i];
        fcr.Dir[dLen] = 0;
        break;
      case      IPSIZE_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) fcr.IPSize[i] = buf[i];
        fcr.IPSize[dLen] = 0;
        break;
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else if (gTag == PIXEL_GROUP) {
    switch (eTag) {
      case      IMSIZE_ELEMENT:
        ImageSize = dLen;
        fcr.HeaderSize = HeaderSize;
        delete buf;
        return fcr;     /* RETURN TO MAIN PROGRAM */
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else {
    /*-------------------------------------------------
          Skip Group
    -------------------------------------------------*/
    fread(buf, sizeof(char), dLen, fp);
    for (i = 0; i < dLen; i++) FcrDcmHeader[HeaderSize + i] = buf[i];
    HeaderSize = HeaderSize + dLen;
    }
  }     /* End of while */
printf("Wrong DICOM Header\n"); exit(0);
}

/*================================================================
        Function getThvDicomInfo()
================================================================*/
struct ThvDicom getThvDicomInfo(FILE *fp, char *THVHeader)
{
int     i;
int     ns, nl;
char    *buf;
FILE    *fq;

unsigned short  gTag, eTag;
unsigned long   dLen, gLen;

unsigned short  getShortInt(char*);
unsigned long   getLongInt(char*);
void            elimSpace(char*, unsigned long*);
void            ShortCutTime(char*);

char    PatBDate[CLEN];         /* Patient Birth Date */
char    PatSex[CLEN];           /* Patient Sex */
int     ImageSize;              /* Image Size (Bytes) */
int     HeaderSize;             // Header Size(Bytes)

struct ThvDicom thv;
```

```
/*------------------------------------------------------------
        Begin
------------------------------------------------------------*/
ns = 2; nl = 4; HeaderSize = 0;
thv.Date[0] = thv.Time[0] = thv.PatName[0] = thv.PatID[0] = 0;

buf = new char[MAX_THV_HEADER_SIZE];

/*------------------------------------------------------------
        Read DICOM Header
------------------------------------------------------------*/
while (ns == (int)fread(buf, sizeof(char), ns, fp)) {
  for (i = 0; i < ns; i++) THVHeader[HeaderSize + i] = buf[i];
  HeaderSize = HeaderSize + ns;
  gTag = getShortInt(buf);
  fread(buf, sizeof(char), ns, fp);
  for (i = 0; i < ns; i++) THVHeader[HeaderSize + i] = buf[i];
  HeaderSize = HeaderSize + ns;
  eTag = getShortInt(buf);
  fread(buf, sizeof(char), nl, fp);
  for (i = 0; i < nl; i++) THVHeader[HeaderSize + i] = buf[i];
  HeaderSize = HeaderSize + nl;
  dLen = getLongInt(buf);

  if    (gTag == STUDY_GROUP) {
    switch (eTag) {
      case      DATE_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) thv.Date[i] = buf[i];
        thv.Date[dLen] = 0;
        break;
      case      TIME_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) thv.Time[i] = buf[i];
        thv.Time[dLen] = 0;
        elimSpace(thv.Time, &dLen);
        ShortCutTime(thv.Time);
        break;
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else if (gTag == PATIENT_GROUP) {
    switch (eTag) {
      case      NAME_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) thv.PatName[i] = buf[i];
        thv.PatName[dLen] = 0;
        elimSpace(thv.PatName, &dLen);
        break;
      case      ID_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
```

```
        for (i = 0; i < dLen; i++) thv.PatID[i] = buf[i];
        thv.PatID[dLen] = 0;
        elimSpace(thv.PatID, &dLen);
        break;
      case      BIRTH_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) PatBDate[i] = buf[i];
        PatBDate[dLen] = 0;
        break;
      case      SEX_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) PatSex[i] = buf[i];
        PatSex[dLen] = 0;
        break;
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else if (gTag == CRSERIES_GROUP) {
    switch (eTag) {
      case      VIEW_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        for (i = 0; i < dLen; i++) thv.Dir[i] = buf[i];
        thv.Dir[dLen] = 0;
        break;
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else if (gTag == IMAGE_GROUP) {
    switch (eTag) {
      case      ROWS_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        thv.Row = (int)getShortInt(buf);
        break;
      case      COLUMNS_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        thv.Col = (int)getShortInt(buf);
        break;
      case      DEPTH_ELEMENT:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
        thv.NBit = (int)getShortInt(buf);
        break;
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
```

```
      }
    }
  else if (gTag == PIXEL_GROUP) {
    switch (eTag) {
      case      IMSIZE_ELEMENT:
        ImageSize = dLen;
        thv.HeaderSize = HeaderSize;
        delete buf;
        return thv;     /* RETURN TO MAIN PROGRAM */
      default:
        fread(buf, sizeof(char), dLen, fp);
        for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
        HeaderSize = HeaderSize + dLen;
      }
    }
  else {
    /*-------------------------------------------------
          Skip Group
    -------------------------------------------------*/
    fread(buf, sizeof(char), dLen, fp);
    for (i = 0; i < dLen; i++) THVHeader[HeaderSize + i] = buf[i];
    HeaderSize = HeaderSize + dLen;
    }
  }     /* End of while */
printf("Wrong DICOM Header\n"); exit(0);
}

/*================================================================
        Function putFcrDicomInfo()
        You can change  (1) Exam date (date and time),
                        (2) Patient's name,
                        (3) Patient's ID number,
                        (4) IP Scanning direction,
                        (5) IP Size,
                        (6) Number of rows,
                        (7) Number of columns,
                        (8) Number of pixels.
================================================================*/
void putFcrDicomInfo(char *FcrDcmHeader, FcrDicom dcm)
{
int             i;
char            buf[32];
int             nbyte = dcm.Col * dcm.Row * sizeof(short);

unsigned short  gTag, eTag;
unsigned long   dLen;

unsigned long   ph = 0;         // Pointer of FcrDcmHeader
int             ns = 2;
int             nl = 4;

unsigned short  getShortInt(char*);
unsigned long   getLongInt(char*);
void    putShortInt(char*, unsigned long, unsigned long, unsigned short);
void    putLongInt (char*, unsigned long, unsigned long, unsigned long);
void    putCharStr (char*, unsigned long, unsigned long, char*);

/*------------------------------------------------------------------
        Modify DICOM Header Information
------------------------------------------------------------------*/
while (ph < dcm.HeaderSize) {
  for (i = 0; i < ns; i++) buf[i] = FcrDcmHeader[ph + i];
  gTag = getShortInt(buf); ph = ph + ns;
  for (i = 0; i < ns; i++) buf[i] = FcrDcmHeader[ph + i];
  eTag = getShortInt(buf); ph = ph + ns;
  for (i = 0; i < nl; i++) buf[i] = FcrDcmHeader[ph + i];
  dLen = getLongInt(buf);  ph = ph + nl;

  if    (gTag == STUDY_GROUP) {
    switch (eTag) {
      case      DATE_ELEMENT:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.Date); break;
      case      DATE_ELEMENT1:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.Date); break;
      case      DATE_ELEMENT2:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.Date); break;
      case      TIME_ELEMENT:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.Time); break;
      case      TIME_ELEMENT1:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.Time); break;
      case      TIME_ELEMENT2:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.Time);
      }
    ph = ph + dLen;
    }
  else if (gTag == PATIENT_GROUP) {
    switch (eTag) {
      case      NAME_ELEMENT:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.PatName); break;
      case      ID_ELEMENT:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.PatID);
      }
    ph = ph + dLen;
    }
  else if (gTag == IP_GROUP) {
    switch (eTag) {
      case      SCAN_ELEMENT:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.Dir); break;
      case      IPSIZE_ELEMENT:
        putCharStr(FcrDcmHeader, ph, dLen, dcm.IPSize);
      }
    ph = ph + dLen;
    }
  else if (gTag == IMAGE_GROUP) {
    switch (eTag) {
      case      ROWS_ELEMENT:
        putShortInt(FcrDcmHeader, ph, dLen, (unsigned short)dcm.Row); break;
      case      COLUMNS_ELEMENT:
        putShortInt(FcrDcmHeader, ph, dLen, (unsigned short)dcm.Col);
      }
    ph = ph + dLen;
    }
  else if (gTag == PIXEL_GROUP) {
    if (eTag == IMSIZE_ELEMENT) {
      ph = ph - nl;
      putLongInt(FcrDcmHeader, ph, (unsigned long)nl, (unsigned long)nbyte);
      ph = ph + nl; break;
      }
    ph = ph + dLen;
    }
  else {
    ph = ph + dLen;
    }
  }
}
```

```
/*=====================================================================
        Function putThvDicomInfo()
=====================================================================*/
void putThvDicomInfo(char *THVHeader, ThvDicom thv)
{
int             i;
char            buf[32];
int             nbyte = thv.Col * thv.Row * sizeof(short);

unsigned short  gTag, eTag;
unsigned long   dLen;

unsigned long   ph = 0;         // Pointer of THVHeader
int             ns = 2;
int             nl = 4;

unsigned short  getShortInt(char*);
unsigned long   getLongInt(char*);
void    putShortInt(char*, unsigned long, unsigned long, unsigned short);
void    putLongInt (char*, unsigned long, unsigned long, unsigned long);
void    putCharStr (char*, unsigned long, unsigned long, char*);

/*---------------------------------------------------------------------
        Modify DICOM Header Information
---------------------------------------------------------------------*/
while (ph < thv.HeaderSize) {
  for (i = 0; i < ns; i++) buf[i] = THVHeader[ph + i];
  gTag = getShortInt(buf); ph = ph + ns;
  for (i = 0; i < ns; i++) buf[i] = THVHeader[ph + i];
  eTag = getShortInt(buf); ph = ph + ns;
  for (i = 0; i < nl; i++) buf[i] = THVHeader[ph + i];
  dLen = getLongInt(buf);  ph = ph + nl;

  if    (gTag == STUDY_GROUP) {
    switch (eTag) {
      case      DATE_ELEMENT:
        putCharStr(THVHeader, ph, dLen, thv.Date); break;
      case      DATE_ELEMENT1:
        putCharStr(THVHeader, ph, dLen, thv.Date); break;
      case      DATE_ELEMENT2:
        putCharStr(THVHeader, ph, dLen, thv.Date); break;
      case      TIME_ELEMENT:
        putCharStr(THVHeader, ph, dLen, thv.Time); break;
      case      TIME_ELEMENT1:
        putCharStr(THVHeader, ph, dLen, thv.Time); break;
      case      TIME_ELEMENT2:
        putCharStr(THVHeader, ph, dLen, thv.Time);
      }
    ph = ph + dLen;
    }
  else if (gTag == CRSERIES_GROUP) {
    if (eTag == VIEW_ELEMENT) putCharStr(THVHeader, ph, dLen, thv.Dir);
    ph = ph + dLen;
    }
  else if (gTag == IMAGE_GROUP) {
    switch (eTag) {
      case      ROWS_ELEMENT:
        putShortInt(THVHeader, ph, dLen, (unsigned short)thv.Row); break;
      case      COLUMNS_ELEMENT:
        putShortInt(THVHeader, ph, dLen, (unsigned short)thv.Col); break;
      case      DEPTH_ELEMENT:
        putShortInt(THVHeader, ph, dLen, (unsigned short)thv.NBit); break;
      case      DEPTH_ELEMENT1:
```

```
        putShortInt(THVHeader, ph, dLen, (unsigned short)(thv.NBit - 1));
      }
    ph = ph + dLen;
    }
  else if (gTag == PIXEL_GROUP) {
    if (eTag == IMSIZE_ELEMENT) {
      ph = ph - nl;
      putLongInt(THVHeader, ph, (unsigned long)nl, (unsigned long)nbyte);
      ph = ph + nl; break;
      }
    ph = ph + dLen;
    }
  else {
    ph = ph + dLen;
    }
  }
}
```

```
        SUBROUTINE EPCT2(IP,JP,ISX,ISY,IXS,IXE,IYS,IYE,NC,IPROC,ITIME)
C
CS    CALL EPCT2(IP,JP,ISX,ISY,IXS,IXE,IYS,IYE,NC,IPROC,ITIME)
C
CP  *****   EXPANSION @ CONTRACTION   **********************************
C
CK        EXPANSION, CONTRACTION, BINARY PICTURE
C
CD          AUG  25, 1973, PROGRAMMED BY J.TORIWAKI
CD          JUNE  7, 1975, REVISED    BY J.TORIWAKI
CD          JUNE  7, 1979, REVISED    BY M.TANAKA
CD          SEPT 20, 1979, SPIDERED   BY M.TEZUKA
CD          JULY 19, 1980, REFORMED   BY E.UENO
C
CA  *****   IP(ISX,ISY)= INPUT BINARY PICTURE
CA  *****   JP(ISX,ISY)= OUTPUT BINARY PICTURE
CA  *****   ISX,ISY=HAIRETSU KP NO OOKISA
CA  *****   IXS,IXE,IYS,IYE=  IX=IXS---IXE,  IY=IYS---IYE NO  HANI NOMI
CA  *****   SYORI SURU
CA  *****   IPROC=0-----CONTRACTION
CA  *****         1-----EXPANSION
CA  *****    ITIME=EXPANSION MATAWA CONTRACTION NO KAISU
CA  *****    NC=RENKETSU-SEI NO SHITEI  =4---4-RENKETSU,
CA  *****                                8---8-RENKETSU.
C
CN  *****   BINARY PATTERN NOMI NI TSUKAERU     ***********-----------
CN          THIS ROUTINE CALLS EXNB
C
C
        implicit integer*4 (i-n)
        INTEGER*4  IP(ISX,ISY)
        INTEGER*4  JP(ISX,ISY)
C
C
        DO 100 IY=1,ISY
        DO 100 IX=1,ISX
            JP(IX,IY)=IP(IX,IY)
100     CONTINUE
C
        JSX=ISX
        JSY=ISY
C
        IF(IPROC.NE.0) GO TO 1000
C
C       **** CONTRACTION *****************************************************
        DO 10 II=1,ITIME
            CALL EXNB2(JP,JSX,JSY,IXS,IXE,IYS,IYE,0,1,2,NC)
C
            DO 30 IY=IYS,IYE
            DO 30 IX=IXS,IXE
                IF(JP(IX,IY).EQ.2) JP(IX,IY)=0
30          CONTINUE
10      CONTINUE
C
        RETURN
C
C       **** EXPANSION *******************************************************
1000    CONTINUE
        DO 20 II=1,ITIME
            CALL EXNB2(JP,JSX,JSY,IXS,IXE,IYS,IYE,1,0,2,NC)
C
            DO 35 IY=IYS,IYE
            DO 35 IX=IXS,IXE
```

```
              IF(JP(IX,IY).EQ.2) JP(IX,IY)=1
35        CONTINUE
20    CONTINUE
C
      RETURN
      END
```

```
        SUBROUTINE EXNB2(KP,ISX,ISY,IX1,IX2,IY1,IY2,N1,N2,N3,NC)
C
CS    CALL EXNB2(KP,ISX,ISY,IX1,IX2,IY1,IY2,N1,N2,N3,NC)
C
CP    *** 2-JIGEN HAIRETU NO ARU-BUBUN RYOOIKI NAI O SEARCH SHITE    ***
CP    *** ARU-ATAI(N2) O MOTU TEN NO KIBAN NO UCHI ARU-ATAI(N1) O    ***
CP    *** MOTU TEN NO ATAI NOMI O ARU-ATAI(N3) NI KAERU.             ***
C
CK        NEIBOUR
CD                         PROGRAMMED BY S.YOKOI
CD                         REVISED    BY J.TORIWAKI
CD          JUNE 29, 1979, REVISED    BY T.HASEGAWA
CD          SEPT 20, 1979, SPIDERED   BY M.TEZUKA
CD          JULY 19, 1980, REFORMED   BY E.UENO
C
CA    *** KP(ISX,ISY) ; INPUT AND OUTPUT                             ***
CA    *** IX1,IX2,IY1,IY2 ; SHORIHANI NO SHITEI                      ***
CA    *** N1,N2 ; HANTEI JYOKEN                                      ***
CA    *** N3 ; DAINYU SURU ATAI                                      ***
CA    *** NC ; DAINYU GIKO SHITEI                                    ***
C
C
        implicit integer*4 (i-n)
        integer*4  KP(ISX,ISY)
C
C
        DO 10 I=IX1,IX2
            DO 20 J=IY1,IY2
                IF(KP(I,J).NE.N2)GO TO 20
                K4=(KP(I-2,J-2)-N1)*(KP(I-1,J-2)-N1)*(KP(I,J-2)-N1)*
     &             (KP(I+1,J-2)-N1)*(KP(I-2,J-2)-N1)*
     &             (KP(I-2,J-1)-N1)*(KP(I-1,J-1)-N1)*(KP(I,J-1)-N1)*
     &             (KP(I+1,J-1)-N1)*(KP(I-2,J-1)-N1)*
     &             (KP(I-2,  J)-N1)*(KP(I-1,  J)-N1)*(KP(I,  J)-N1)*
     &             (KP(I+1,  J)-N1)*(KP(I-2,  J)-N1)*
     &             (KP(I-2,J+1)-N1)*(KP(I-1,J+1)-N1)*(KP(I,J+1)-N1)*
     &             (KP(I+1,J+1)-N1)*(KP(I-2,J+1)-N1)*
     &             (KP(I-2,J+2)-N1)*(KP(I-1,J+2)-N1)*(KP(I,J+2)-N1)*
     &             (KP(I+1,J+2)-N1)*(KP(I-2,J+2)-N1)
                IF(K4.NE.0)GO TO 15
                KP(I,J)=N3
                GO TO 20
15              CONTINUE
20          CONTINUE
10      CONTINUE
C
        RETURN
        END
```

```
/*===============================================================
Name        fcrsubUtil.c
Function    Utility for Reading FCR Standard Header
Author            SHIGE 11/04/98
=================================================================*/
#include    <stdio.h>
#include    <string.h>
#include    <stdlib.h>
#include        <ctype.h>
#include    <math.h>
#include        "TempSub.H"

/*=================================================================
      FcrStand getFcrStandInfo(FILE *fp, char *FCRHeader)
      *fp         File Pointer            [I]
      *FCRHeader  FCR Standard Header     [O]
      Return Value      FcrStand Structure      [O]
      You must allocate FCRHeader before calling this routine.
      Only alphanumeric value is available in fcr.PatName.
===================================================================*/
FcrStand getFcrStandInfo(FILE *fp, char *FCRHeader)
{
FcrStand    fcr;
return fcr;
}
```

```
#include    <stdio.h>
#include    <stdlib.h>
#include    <string.h>
#include    <ctype.h>

#define           LENGTH      30

#ifdef KRL_GNU
  #define fname1_printq fname1_printq__
  #define fname2_printq fname2_printq__
  #define fname_LUT fname_LUT__
#else
  #define fname1_printq fname1_printq_
  #define fname2_printq fname2_printq_
  #define fname_LUT fname_LUT_
#endif

extern "C" void fname1_printq(char *file1, char *file2, char *fname, int
*p_tps, int *p_sas, int *p_inc,
            int L_file1, int L_file2, int L_fname, int L_tps, int L_sas,
int L_inc)
{
      int   length, i;
      char  fname1[LENGTH], fname2[LENGTH];
      int   tps, sas, inc;

      for(i = 0; i < LENGTH; i++)   {
            fname1[i] = 0;    fname2[i] = 0;
      }

      length = LENGTH;
      for( i = 0; i < LENGTH; i++) {
            if(isalnum(file1[i]) == 0) {
                  length = i;
                  break;
            }
      }
      strncpy(fname1, file1, length);

      length = LENGTH;
      for( i = 0; i < LENGTH; i++) {
            if(isalnum(file2[i]) == 0) {
                  length = i;
                  break;
            }
      }
      strncpy(fname2, file2, length);

      tps = *p_tps, sas = *p_sas, inc = *p_inc;

      sprintf(fname, "%s-%s_T%iS%iI%i_Q.DAT",
            fname2, fname1, tps, sas, inc );
}

extern "C" void fname2_printq(char file1[], char file2[], char fname[],
int *p_tps, int *p_sas, int *p_inc, int *p_order,
            int L_file1, int L_file2, int L_fname, int L_tps, int L_sas,
int L_inc, int L_order)
{
      char  fname1[LENGTH], fname2[LENGTH];
      int   tps, sas, inc, order, i, length;
/*    initialization    */
      for(i = 0; i < LENGTH; i++)    {
            fname1[i] = 0;    fname2[i] = 0;
      }

/*    find the space character      */
      length = LENGTH;
      for( i = 0; i < LENGTH; i++) {
            if(isalnum(file1[i]) == 0) {
                  length = i;
                  break;
            }
      }
/*    copy file1 to fname1 till the space character   */
      strncpy(fname1, file1, length);

      length = LENGTH;
      for( i = 0; i < LENGTH; i++) {
            if(isalnum(file2[i]) == 0) {
                  length = i;
                  break;
            }
      }
      strncpy(fname2, file2, length);

      tps = *p_tps, sas = *p_sas, inc = *p_inc, order = *p_order;

      sprintf(fname, "%s-%s_T%iS%iI%i_F%i_Q.DAT",
            fname2, fname1, tps, sas, inc, order );
}


extern "C" void fname_LUT(char *head, char *dirlut, int L_head, int
L_dirlut)
{
      char  head_of_dir[60];
      int   i, length;

      for( i = 0; i < 60; i++)      head_of_dir[i] = 0;

      for( i = 0; i < LENGTH; i++) {
            if(isalnum(head[i]) == 0) {
                  length = i;
                  break;
            }
      }
      strncpy(head_of_dir, head, length);
      sprintf(head, "%sdupect", head_of_dir);
}
```

```
!========================================================================
        subroutine get_density_correction_factor( oriImage, ncol, nlin,
        1               grayscale, dens, type, dirlut, leng, DCTable, TableNo )
!========================================================================
!       Ver. 2.1
!       Written by Xin-Wei Xu,  "DENSITYCORRECTION"
!       Modified by Akiko Kano,  Mar.29, 1993
!       Modified by Shige,       Apr.28, 1999
!       Modified by Shige,       Jun.15, 1999   Add 1.0x LUT
!------------------------------------------------------------------------
!       This subroutine determines the exposure level factor and locates the
!       Look-Up-Table for a nonlinear density correction.
!
!       (1) LUTs are suitable for grayscale ranging from 0 to 1023, in which "0"
!           corresponds to high optical density and "1023" corresponds to low
!           optical density.
!       (2) If "grayscale" is 1, which means that "0" corresponds to low optical
!           density, this subroutine returns image data with the inverted gray
!           scale for later use.
!========================================================================
!       ARGUMENTS
!------------------------------------------------------------------------
        implicit integer*2 (i-n)

        integer*4 ncol, nlin           ! Matrix Size of Image Data        [I]
        integer*2 oriImage(ncol,nlin)  ! Input and Output Image Data    [I,O]
        integer*4 grayscale            ! 1:0->Lighter, -1:0->Darker       [I]
        integer*2 dens                 ! Scanner Density Range            [I]
                                       !  3:(0-3)4:(0-4)
        integer*2 type                 ! Screen/Film Type                 [I]
                                       !  1:Med/OC, 2:Med/TMG, 3:insight HC
        character dirlut*(*)           ! Subdirectory Where LUTs are Stored[I]
        integer*4 leng                 ! Length of dirlut                 [I]
        integer*2 DCTable(2,1024)      ! LUT for Density Correction       [O]
                                       !  DCTable(1,*):Input Pixel Value
                                       !  DCTable(2,*):Output Pixel Value
        integer*2 TableNo              ! LUT No. for Density Correction   [O]
                                       ! 1:0.25x   2:0.30x   3:0.35x   4:0.42x
                                       ! 5:0.50x   6:0.60x   7:0.71x   8:0.84x
                                       ! 9:1.00x
                                       !10:1.19x  11:1.41x  12:1.68x  13:2.00x
                                       !14:2.37x  15:2.82x  16:3.35x  17:3.98x
!------------------------------------------------------------------------
!       VARIABLES
!------------------------------------------------------------------------
        integer*2 LUT(1024,17),
     &            ECT_NOR(17),
     &            ECT_UND(8),
     &            ECT_OVE(8),ECT_OVE1(8),ECT_OVE2(8),ECT_OVE3(8),
     &            P25,       ! pixel value at 25% from minimum of histogram
                             ! at center quarter ROI of chest image
     &            Pmin,      ! minimum pixel value of the Histogram
     &            expoIndex, !=0:Normal expo; =1:over expo; =2:under expo;
     &            Di, Di1,
     &            nxw,nyh
        integer Hist(0:1023), ! histogram of center quarter ROI of chest image
     &          total         ! total pixel number of the histogram
        real    fraction(0:1023) ! histogram fraction
        character*80 dcTableName  !density correction look up table name
        character*5 expolevel(17)
        integer*2       ngray
c***********************************************************************
        data expolevel/'0.25x','0.30x','0.35x','0.42x',
```

```
     &                 '0.50x','0.60x','0.71x','0.84x',
     &                 '1.00x',
     &                 '1.19x','1.41x','1.68x','2.00x',
     &                 '2.37x','2.82x','3.35x','3.98x'/
c************************************************************************
!
!------------------------------------------------------------------------
!       BEGIN
!------------------------------------------------------------------------
c
c  part 0: Initialize
c
c************************************************************************
        nxw = ncol
        nyh = nlin

        if ( grayscale.eq.1 ) then
            do iy = 1, nyh
                do ix = 1, nxw
                    oriimage(ix,iy) = 1023 - oriimage(ix,iy)
                end do
            end do
        end if

c************************************************************************
c
c  part 1: make a look up table matrix "LUT" for later to determine
c          a proper look up table to correct exposure level of images
c
c************************************************************************

        call MakeLUT(dens,type,LUT,dirlut,leng)

c ** end of part 1 **

c***********************************************************************
c
c  part 2: obtain histogram of the image in center quarter ROI
c
c***********************************************************************

        do i=0,1023
          Hist(i)=0
          fraction(i)=0.0
        end do

        nxq=int(float(nxw)/4.0)
        nyq=int(float(nyh)/4.0)

        do iy=nyq, 3*nyq
          do ix=nxq, 3*nxq
            Hist(oriImage(ix,iy))=Hist(oriImage(ix,iy))+1
          end do
        end do

c ** end of part 2 **

c***********************************************************************
c
c  part 3: analyze the histogram to obtain P25 and Pmin
c
c***********************************************************************
```

```
        do i=0, 1023
          if (Hist(i).gt.0) then
            inimin=i
            go to 5
          end if
        end do
5       continue

        do i=1023, 0, -1
          if (Hist(i).gt.0) then
            inimax=i
            go to 10
          end if
        end do
10      continue

        total=0
        do i=inimin, inimax
          total=total+Hist(i)
        end do
        ftotal=float(total)

        do i=inimin, inimax
          fraction(i)=100.0*(float(Hist(i)))/ftotal
        end do
        do i=0, inimin-1
          fraction(i)=0.0
        end do
        do i=inimax+1, 1023
          fraction(i)=0.0
        end do

        percent25=0.0
        do i=inimin, inimax
          percent25=percent25+fraction(i)
          if (percent25.ge.25.0) then
            P25=i
            go to 15
          end if
        end do
15      continue    ! determine P25

        do i=P25, inimin, -1
          if (fraction(i).le.0.01) then
            Pmin=i
            go to 20
          end if
        end do
20      continue    ! determine Pmin

c       type*,' P25=',P25,'      Pmin=',Pmin

c** end of part 3 **

c***********************************************************************
c
c  part 4 determine exposure contion of chest images by its Pmin
c
c***********************************************************************

        if (dens.eq.3) then
          if (Pmin.lt.180) then
            expoIndex=1
```

```
           else if (Pmin.gt.500) then
            expoIndex=2
           else
            expoIndex=0
          end if
        end if

        if (dens.eq.4) then
          if (Pmin.lt.396) then
            expoIndex=1
           else if (Pmin.gt.633) then
            expoIndex=2
           else
            expoIndex=0
          end if
        end if

c       if (expoIndex.eq.0) then
c         type*,'The image is within normal exposure range'
c        else if (expoIndex.eq.1) then
c         type*,'The image is over exposed'
c        else
c         type*,'The image is under exposed'
c       end if

c ** end of part 4 **

c***********************************************************************
c
c   part 5: set density correction level
c
c***********************************************************************

        if (dens.eq.3) then
          level_nor=350
          level_und=350
          level_ove1=400
          level_ove2=350
          level_ove3=300
        end if

        if (dens.eq.4) then
          level_nor=522
          level_und=522
          level_ove1=558
          level_ove2=522
          level_ove3=485
        end if

        do j=1, 8
          ECT_UND(j)=0
          ECT_OVE1(j)=0
          ECT_OVE2(j)=0
          ECT_OVE3(j)=0
        end do
        do j=1, 17
          ECT_NOR(j)=0
        end do

        do j=1, 8
          do i=1,1024
            if ((LUT(i,j).le.level_und).and.(LUT(i+1,j).gt.level_und))
     &        then
```

```
              ECT_UND(j)=i
            end if
          end do
        end do

        do j=10, 17
          j8=j-9

          do i=1,1024
            if ((LUT(i,j).le.level_ove1).and.(LUT(i+1,j).gt.level_ove1))
     &        then
                ECT_OVE1(j8)=i
            end if
          end do

          do i=1,1024
            if ((LUT(i,j).le.level_ove2).and.(LUT(i+1,j).gt.level_ove2))
     &        then
                ECT_OVE2(j8)=i
            end if
          end do

          do i=1,1024
            if ((LUT(i,j).le.level_ove3).and.(LUT(i+1,j).gt.level_ove3))
     &        then
                ECT_OVE3(j8)=i
            end if
          end do

        end do

        do j=1,17
          do i=1, 1024
            if ((LUT(i,j).le.level_nor).and.(LUT(i+1,j).gt.level_nor))
     &        then
                ECT_NOR(j)=i
            end if
          end do
        end do

c*****************************************************************************
c
c       type*,'ECT_UND(crosspoints)'
c       do k=1,8
c         type*,'*',k,ECT_UND(k)
c       end do
c       type*,'ECT_OVE1(high level(1) crosspoints)'
c       do k=1,8
c         type*,'*',k,ECT_OVE1(k)
c       end do
c       type*,'ECT_OVE2(mid level(2) crosspoints)'
c       do k=1,8
c         type*,'*',k,ECT_OVE2(k)
c       end do
c       type*,'ECT_OVE3(low level(3) crosspoints)'
c       do k=1,8
c         type*,'*',k,ECT_OVE3(k)
c       end do
c       type*,'ECT_NOR(crosspoints)'
c       do k=1, 16
c         type*,'*',k,ECT_NOR(k)
c       end do
c
```

```
c*****************************************************************************

c ** end of part 5 **

c*****************************************************************************
c
c   part 6: determine the proper look up table for density correction
c                             (TableNo)
c
c*****************************************************************************
        TableNo = 0

      if (expoIndex.eq.0) then
        do i=1, 17
          if ((Pmin.le.ECT_NOR(i)).and.(Pmin.ge.ECT_NOR(i+1))) then
              Di=ECT_NOR(i)-Pmin
              Di1=Pmin-ECT_NOR(i+1)
c             if (Di.eq.Di1) go to 100
              if (Di.eq.Di1.and.i.eq.8) go to 100 ! no density correction
c             if (Di.lt.Di1) then
              if (Di.le.Di1) then
                TableNo=i
               else
                TableNo=i+1
              end if
              go to 30
          end if
        end do

       else if (expoIndex.eq.1) then
        if (dens.eq.3) then
          if (P25.lt.(Pmin+200)) then
            do i=1, 8
              ECT_OVE(i)=ECT_OVE1(i)
            end do
           else if (P25.gt.(Pmin+350)) then
            do i=1, 8
              ECT_OVE(i)=ECT_OVE3(i)
            end do
           else
            do i=1, 8
              ECT_OVE(i)=ECT_OVE2(i)
            end do
          end if
        end if

        if (dens.eq.4) then
          if (P25.lt.(Pmin+412)) then
            do i=1, 8
              ECT_OVE(i)=ECT_OVE1(i)
            end do
           else if (P25.gt.(Pmin+522)) then
            do i=1, 8
              ECT_OVE(i)=ECT_OVE3(i)
            end do
           else
            do i=1, 8
              ECT_OVE(i)=ECT_OVE2(i)
            end do
          end if
        end if

        if (Pmin.le.ECT_OVE(8)) then
```

```
          TableNo=17
         else if (Pmin.ge.ECT_OVE(1)) then
          TableNo=10
         else
          do i=1, 8
            if ((Pmin.le.ECT_OVE(i)).and.(Pmin.ge.ECT_OVE(i+1))) then
              Di=ECT_OVE(i)-Pmin
              Di1=Pmin-ECT_OVE(i+1)
              if (Di.ge.Di1) then
                TableNo=i+1+9
               else
                TableNo=i+9
              end if
              go to 30
            end if
          end do
        end if

       else
        if (Pmin.ge.ECT_UND(1)) then
          TableNo=1
         else if (Pmin.le.ECT_UND(8)) then
          TableNo=8
         else
          do i=1, 8
            if ((Pmin.le.ECT_UND(i)).and.(Pmin.ge.ECT_UND(i+1))) then
              Di=ECT_UND(i)-Pmin
              Di1=Pmin-ECT_UND(i+1)
              if (Di.le.Di1) then
                TableNo=i
               else
                TableNo=i+1
              end if
              go to 30
            end if
          end do
        end if

      end if

30    continue

        write(*,*) ' Exposure level : ', expolevel(TableNo)
c       write(*,*) ' Table No.      : ', TableNo

c ** end of part 6 **

c*****************************************************************************
c
c   part 7: load determined look up table
c
c*****************************************************************************

        call TableName(TableNo,dens,type,dirlut,leng,dcTableName)
        ngray = 1024
        call loadTable(dcTableName,dcTable,ngray)

100     continue

c ** end of part 7 **
!-------------------------------------------------------------------------------
!       END
!-------------------------------------------------------------------------------
```

```
        return
        end
!
c***********************************************
c
c  Obtain look up table name
c
c  Name :TableName(No,dens,type,dirlut,leng,filename)
c
c***********************************************

        subroutine TableName(No,dens,type,dirlut,leng,filename)

        implicit integer*2 (i-n)

        integer*2  No,        !look up table number
     &             dens,      !scanner density range:3(0-3);4(0-4);14(1-4);
     &             type       !screen-film system type: 1(Med/OC);2(Med/TMG);
                              ! 3(insight HC); etc.

        character dirlut*(*)
        integer*4 leng
        integer*4 nchar
        character*128 filename
        character*40 head
        character*10  tail
        character*10  density,
     &                tableNo,
     &                number(17)
c       character*6  tail
c       character*2  density,
c     &                       tableNo,
c     &               number(17)

c***********************************************

        data number/'U8','U7','U6','U5','U4','U3','U2','U1',
     &              'NO',
     &              'O1','O2','O3','O4','O5','O6','O7','O8'/


        head = 'DUPECT'

        if (dens.eq.3) then
          density="03"
        end if
        if (dens.eq.4) then
          density="04"
        end if

        tableNo=number(No)

        if (type.eq.1) then
          tail='C1.DAT'
        end if
        if (type.eq.2) then
          tail='A1.DAT'
        end if

        call nullplus(head,    nchar)
        call nullplus(density, nchar)
        call nullplus(tableNo, nchar)
        call nullplus(tail,    nchar)
        call lut_filename(dirlut, head, density, tableNo, tail, filename)
        call nullminus(filename, nchar)
c       write(*,*) 'LUT Filename = ', nchar, ':', filename

        return
        end

c*********************************************************************
c
c   Make a look up table matrix "LUT"
c
c   Name: MakeLUT(dens,type,LUT,dirlut,leng)
c
c*********************************************************************

        subroutine  MakeLUT(dens,type,LUT,dirlut,leng)

        implicit integer*2 (i-n)

        integer*2
     &            dens,     !scanner density range:3(0-3);4(0-4);14(1-4);
     &            type,     !screen-film system type: 1(Med/OC);2(Med/TMG);
                            ! 3(insight HC); etc.
     &            LUT(1024,17)  !look up table matrix

        integer*2  dcTable(2,1024)
        character  dirlut*(*)
        integer*4  leng
        integer*2  ngray

        character*128 dcTableName

c*************************************************************************

        ngray = 1024
        do No=1, 17
          call TableName(No,dens,type,dirlut,leng,dcTableName)
          call loadTable(dcTableName,dcTable,ngray)
          do j=1, 1024
            LUT(j,No)=dcTable(2,j)
          end do
        end do

        return
        end

c*************************************************************************
c
c   Open a "dc" look up table and put it to a buffer "dcTable"
c
c   Name: loadTable(dcTableName,dcTable,N)
c
c*************************************************************************

        subroutine loadTable(dcTableName,dcTable,N)

        IMPLICIT INTEGER*2 (I-N)

        integer*2 dcTable(2,N)

        character*128 dcTableName
        integer*2       buf(1024)
```

```
        integer*4       nchar
c*************************************************************************
c       open (unit=4, file=dcTableName, status='old')
c       write(*,*) dcTableName
c        do j=1,N
c            read (4,100) dcTable(1,j),dcTable(2,j)
c        end do
c       close (unit=4)
c100     format(2i10)

        call nullplus(dcTableName, nchar)
        call read_table(dcTableName, buf, N)
        do i = 1, N
          dcTable(1,i) = i - 1
          dcTable(2,i) = buf(i)
        end do
        call nullminus(dcTableName, nchar)

        return
        end
```

INITIALIZATION.f

```
!===============================================================================
       subroutine  Initialization( DC1, DC2, blank, ncol, nlin )
!===============================================================================
!      Ver. 1.0
!      Written by Akiko Kano,  Mar.24, 1993
!-------------------------------------------------------------------------------
!      This subroutine initializes some parameters.
!===============================================================================
!      ARGUMENTS
!-------------------------------------------------------------------------------
       implicit  none
c      integer*4 Initialization
       integer*4 ncol, nlin             ! Matrix Size of Image Data          [I]
       integer*4 DC1, DC2               ! 1 -> Dens. Corr. is Done         [I,O]
     1                                  ! 0 -> Dens. Corr. is Not Done
       integer*2 blank(ncol, nlin)      ! 1 -> Pixel with No Image Data    [I,O]
                                        ! 0 -> Pixel with Image Data
!-------------------------------------------------------------------------------
!      VARIABLES
!-------------------------------------------------------------------------------
       integer*4 C, L
!-------------------------------------------------------------------------------
!      BEGIN
!-------------------------------------------------------------------------------
       DC1 = 0
       DC2 = 0

       do L = 1, nlin
           do C = 1, ncol
               blank(C,L) = 0
           end do
       end do
!-------------------------------------------------------------------------------
!      END
!-------------------------------------------------------------------------------
       return
       end
```

```
!       ---------------------------------------------------------------
!       Initial image matching
!       ---------------------------------------------------------------
        subroutine
init_match(iwork_1_orig,iwork_2_orig,mszx_orig,mszy_orig,rribcage1_orig,
     &          lribcage1_orig,rribcage2_orig,lribcage2_orig,
     &          ribfeature1_orig,ribfeature2_orig,
     &          rribcage_no1_orig,lribcage_no1_orig,rribcage_no2_orig,
     &          lribcage_no2_orig,
     &          ixshift,iyshift,cmax,mszx146,mszy146,img1146,img2146,
     &          out_img1,out_img2,out_img1wk,out_img2wk,out_img1c,
     &          out_img2c)

!-----------------------------------------------------------------------
!       Modified by Qiang Li, enable it to process images with any matrix size
!       Oct. 30, 1998
!-----------------------------------------------------------------------
!       Function of this subroutine is to match images globally by using a
!       cross-correlation technuque.
!
!       (1) Lateral inclination should be corrected in advance.
!       (2) First, the original images (a current and a previous image) are
!           reduced to 146 x 146 (1/4 of FCR image) by using averaging.
!           (Original FCR image matrix size is 1760 x 1760.)
!       (3) Then, these low-resolution images are blurred with a Gaussian
!           filter (9 x 9 mask size).
!       (4) A lung segmentation by use of the detected ribcage edges is
!           employed for elimination of unwanted area, that is, outside
!           of ribcage edges and below diaphragms.
!       (5) We employed cross-correlation method for matching of the blurred
!           low-resolution images obtained in this subroutine.
!       (6) Cross-correlation values are obtained to determine the horizontal
!           and vertical shift-values between the two images.
!       (7) Thus, the shift-values are obtained for the global matching.
!
!       ---------------------------------------------------------------
        implicit integer*4 (i-n)

        parameter  (mszx=586)            ! Marix size of the current image.(column)
        parameter  (mszy=586)            ! Marix size of the current image.(line)
        parameter (mszxtmp=100)          ! Width of the template image
        parameter (mszytmp=60)           ! Hight of the template image


        integer*2  iwork_1_orig(mszx_orig,mszy_orig)            ! current image
        integer*2  iwork_2_orig(mszx_orig,mszy_orig)

        integer*2 iwork1(mszx,mszy)      ! Original current image size          [I]
        integer*2 iwork2(mszx,mszy)      ! Original previous image size         [I]
c       integer*4 mszx,mszy              ! matrix size of original image        [I]
                                         ! (Default size = 586 x 586)
        integer*2 rribcage1_orig(2,1215)         ! Right Ribcage Points for Image1
[I]
        integer*2 lribcage1_orig(2,1215)         ! Left Ribcage Points for Image1
[I]
        integer*2 rribcage2_orig(2,1215)         ! Right Ribcage Points for Image2
[I]
        integer*2 lribcage2_orig(2,1215)         ! Left Ribcage Points for Image2
[I]
        integer*2 ribfeature1_orig(50)   ! Ribcage-Related Features for Image1  [I]
        integer*2 ribfeature2_orig(50)   ! Ribcage-Related Features for Image2  [I]
        integer*2 rribcage_no1_orig      ! No. of Right Ribcage Points for Image1
[I]
```

```
        integer*2 lribcage_no1_orig      ! No. of Left Ribcage Points for Image1
[I]
        integer*2 rribcage_no2_orig      ! No. of Right Ribcage Points for Image2
[I]
        integer*2 lribcage_no2_orig      ! No. of Left Ribcage Points for Image2
[O]

        integer*2 rribcage1(2,1215)      ! Right Ribcage Points for Image1         [I]
        integer*2 lribcage1(2,1215)      ! Left Ribcage Points for Image1          [I]
        integer*2 rribcage2(2,1215)      ! Right Ribcage Points for Image2         [I]
        integer*2 lribcage2(2,1215)      ! Left Ribcage Points for Image2          [I]
        integer*2 ribfeature1(50)        ! Ribcage-Related Features for Image1     [I]
        integer*2 ribfeature2(50)        ! Ribcage-Related Features for Image2     [I]
        integer*2 rribcage_no1   ! No. of Right Ribcage Points for Image1          [I]
        integer*2 lribcage_no1   ! No. of Left Ribcage Points for Image1           [I]
        integer*2 rribcage_no2   ! No. of Right Ribcage Points for Image2          [I]
        integer*2 lribcage_no2   ! No. of Left Ribcage Points for Image2           [O]
!       ---------------------------------------------------------------------------
!       ixshift,iyshift are the most important outputs.
!       ---------------------------------------------------------------------------
        integer*4 ixshift,iyshift        ! Global shift values for x and y         [O]
!       ---------------------------------------------------------------------------
!       cmax is not used in the main routine. This could be eliminated.
!       ---------------------------------------------------------------------------
        real*4    cmax                   ! Cross-correlation value at the best
                                         ! fit point.                              [O]
        integer*4 mszx146,mszy146        ! matrix size of reduced image            [I]
                                         ! (default size = 146 x 146)
        integer*2 img1146(mszx146,mszy146)       ! Averaged low-resolution current
                                                 ! image.                          [O]
        integer*2 img2146(mszx146,mszy146)       ! Averaged low-resolution previous
                                                 ! image.                          [O]
        integer*2 out_img1(mszx146,mszy146) !Gaussian filtered current image       [O]
        integer*2 out_img2(mszx146,mszy146) !Gaussian filtered previous image      [O]
        integer*2 out_img1wk(mszx146,mszy146)    ! Copy of Gaussian filtered current
                                                 ! image. (Work image)             [O]
        integer*2 out_img2wk(mszx146,mszy146)    ! Copy of Gaussian filtered previous

                                                 ! image. (Work image)             [O]
        integer*2 out_img1c(mszx146,mszy146)     ! Lung segmentated current image[O]
        integer*2 out_img2c(mszx146,mszy146)     ! Lung segmentated previous image[O]
        integer*2 temp(mszxtmp,mszytmp)          ! Template image for
cross-correlation
                                                 ! (Upper 100 x 60 matrix area of
                                                 ! the previous image).

        real*4  gauss(9,9)                       ! Gaussian filter (9 x 9 matrix
size)
!       ---------------------------------------------------------------------------
!       "corr_mat" is a cross-correlation values for each point to check the
program.
!       Not important.
!       ---------------------------------------------------------------------------
c       real*4  corr_mat(30,30)                  ! Cross-correlation values
        real*4  corr_mat(50,50)                  ! Cross-correlation values

        integer*4 isa(4)                         ! Location for search area of
                                                 ! cross-correlation.
                                         ! isa(1),isa(2): x, y location for
upper-left
                                         ! isa(3),isa(4): x, y location for
lower-right
!******************************************************************************
```

```
!******************************************************************************
c       ---------------------------------------------------------------
c       enlarge/reduce the size of image to 586*586
c       because the ORIGIANL subroutine is designed to process this image size
C       Modified by Qiang Li
c       ---------------------------------------------------------------

        call Image_Scaling(iwork_1_orig, mszx_orig, mszy_orig, iwork1, mszx, mszy)
        call Image_Scaling(iwork_2_orig, mszx_orig, mszy_orig, iwork2, mszx, mszy)

        rribcage_no1 = rribcage_no1_orig
        lribcage_no1 = lribcage_no1_orig
        do i=1,rribcage_no1
           rribcage1(1,i) = rribcage1_orig(1,i)
           rribcage1(2,i) = rribcage1_orig(2,i)
        end do
        do i=1,lribcage_no1
           lribcage1(1,i) = lribcage1_orig(1,i)
           lribcage1(2,i) = lribcage1_orig(2,i)
        end do
        do i=1,16
           ribfeature1(i) = ribfeature1_orig(i)
        end do

        call Curve_Scaling(rribcage1, rribcage_no1, float(mszx) / float(mszx_orig))
        call Curve_Scaling(lribcage1, lribcage_no1, float(mszx) / float(mszx_orig))
        call Rib_Feature_Scaling(ribfeature1, float(mszx) / float(mszx_orig) )

        rribcage_no2 = rribcage_no2_orig
        lribcage_no2 = lribcage_no2_orig
        do i=1,rribcage_no2
           rribcage2(1,i) = rribcage2_orig(1,i)
           rribcage2(2,i) = rribcage2_orig(2,i)
        end do
        do i=1,lribcage_no2
           lribcage2(1,i) = lribcage2_orig(1,i)
           lribcage2(2,i) = lribcage2_orig(2,i)
        end do
        do i=1,16
           ribfeature2(i) = ribfeature2_orig(i)
        end do

        call Curve_Scaling(rribcage2, rribcage_no2, float(mszx) / float(mszx_orig))
        call Curve_Scaling(lribcage2, lribcage_no2, float(mszx) / float(mszx_orig))
        call Rib_Feature_Scaling(ribfeature2, float(mszx) / float(mszx_orig) )

c       ---------------------------------------------------------------
c       0 set for Template image
c       ---------------------------------------------------------------
        do j=1,mszytmp
            do i=1,mszxtmp
                temp(i,j)=0
            end do
        end do
c       ---------------------------------------------------------------
c       REDUCE IMAGE BY AVERAGING mszx,mszy ---> mszx146,mszy146
c       ---------------------------------------------------------------
        istep=4                          ! reducing factor (4 means 1/4)
        avgres1=0                        ! work variable for averaging of current
image
        avgres2=0                        ! work variable for averaging of previous
image
```

```
        icty=0
        do j=1,mszy,istep
                icty=icty+1
                ictx=0
                do i=1,mszx,istep
                        ictx=ictx+1
                        itotal1=0
                        itotal2=0
                        do m=j,j+istep-1
                        do l=i,i+istep-1
                                itotal1=itotal1+iwork1(l,m)
                                itotal2=itotal2+iwork2(l,m)
                        end do
                        end do
                        avgres1=float(itotal1)/float(istep**2)
                        avgres2=float(itotal2)/float(istep**2)
                        img1146(ictx,icty)=avgres1 ! low-resolution current image
                        img2146(ictx,icty)=avgres2 ! low-resolution previous image
                end do
        end do
c       ---------------------------------------------------------------
c       make a Gaussian filter (9 x 9)
c       ---------------------------------------------------------------
        msksz=9                          ! mask size of Gaussian filter
        mskst=msksz/2                    ! center of the filter
        sigma=1.6                        ! parameter (SD) of Gaussian filter

        do j=1,msksz
        do i=1,msksz
                icenx=mskst+1
                iceny=mskst+1
                Gauss(i,j)=1/(2*3.141592*sigma**2)*
     &  exp((-((icenx-i)**2+(iceny-j)**2))/(2*sigma**2))
        end do
        end do

c       do j=1,msksz
c       write(*,9999) (Gauss(i,j),i=1,msksz)
c       end do
c 9999  format (9f5.3)

c       ---------------------------------------------------------------
c       Make blurred low-resolution images
c       ---------------------------------------------------------------
c       ---------------------------------------------------------------
c       0 set of the dimensions
c       ---------------------------------------------------------------
        do j=1,mszy146
        do i=1,mszx146
                out_img1(i,j)=0
                out_img2(i,j)=0
                out_img1wk(i,j)=0
                out_img2wk(i,j)=0
        end do
        end do
c       ---------------------------------------------------------------
c       2. Gaussian filtering for current and previous image.
c       ---------------------------------------------------------------
        do j=1+mskst,mszy146-mskst
            do i=1+mskst,mszx146-mskst
                wk1=0
                wk2=0
                do n=1,msksz
```

```
                 do m=1,msksz
                   wk1= wk1+ img1146(i+m-mskst,j+n-mskst)*Gauss(m,n)
                   wk2= wk2+ img2146(i+m-mskst,j+n-mskst)*Gauss(m,n)
                 end do
              end do
              out_img1(i,j)=int(wk1)
              out_img2(i,j)=int(wk2)
              out_img1wk(i,j)=int(wk1)
              out_img2wk(i,j)=int(wk2)
           end do
      end do
c     ------------------------------------------------------------
c     Average along the ribcage edges(image #1: current image)
c     and determine the Max. and Min. values in the Mid. quadrant
c     area of the current image.
c     Average value along the ribcage edge is used to fill the
c     outside of the ribcage edges.
c     Max. and Min. values in the Mid. quadrant area for current image
c     are used for normalization of the blurred low-resolution images.
c
c     This part is related to the normalization of current image.
c     ------------------------------------------------------------
      avg1=0
      do i=1,rribcage_no1
              ixr=rribcage1(1,i)/istep
              iyr=rribcage1(2,i)/istep
              if(ixr.le.0) ixr=1
              if(iyr.le.0) iyr=1
              avg1=avg1+out_img1(ixr,iyr)
      end do
      do i=1,lribcage_no1
              ixl=lribcage1(1,i)/istep
              iyl=lribcage1(2,i)/istep
              if(ixl.eq.0) ixl=1
              if(iyl.eq.0) iyl=1
              avg1=avg1+out_img1(ixl,iyl)
      end do

      avg1=avg1/float(rribcage_no1+lribcage_no1) ! Average along the ribcage edge
!     ---------------------------------------------------------------
!     If the ribcage edges are mis-detected in the lung area, average
!     pixel value along the ribcage edge could be very low value.
!     To aboid this, minimum value of the avg1 is setted to 500.
c
c     This part is related to the normalization of image.
!     ---------------------------------------------------------------
      if(avg1.lt.500) avg1=500

c     write(*,*) 'Avg. on ribcage edge for current image:',avg1

      max1=-1000                     ! Maximum pixel value in mid. quadrant area
      min1= 1000                     ! Minimum pixel value in mid. quadrant area
      do j=(mszy146/istep)*1,(mszy146/istep)*3
      do i=(mszx146/istep)*1,(mszx146/istep)*3
              if (out_img1(i,j).gt.max1) max1=out_img1(i,j)
              if (out_img1(i,j).lt.min1) min1=out_img1(i,j)
      end do
      end do
c     write(*,*) 'max1,min1 for current image',max1,min1

      ifill=int(avg1)                ! "ifill" is the value for filling into
                                     ! outside of ribcage border.
c     ------------------------------------------------------------
```

```
c     Normalization of the current blurred low-resolution image
c     ---------------------------------------------------------------
      do j=1,mszy146
         do i=1,mszx146
              out_img1(i,j)= (1023/float((ifill-min1)))*out_img1(i,j)
     &        -(1023*min1)/float((ifill-min1))
              if(out_img1(i,j).lt.0) out_img1(i,j)=0
              if(out_img1(i,j).gt.1023) out_img1(i,j)=1023
              out_img1wk(i,j)= (1023/float((ifill-min1)))*out_img1wk(i,j)
     &        -(1023*min1)/float((ifill-min1))
              if(out_img1wk(i,j).lt.0) out_img1wk(i,j)=0
              if(out_img1wk(i,j).gt.1023) out_img1wk(i,j)=1023
          end do
      end do

      ifill=1023                     ! After the low-resolution image
                                     ! is normalized, "ifill" is
                                     ! setted to 1023.
c     -------------------------------------------------------------------
c     Fill rims of the current blurred low-resolution image caused by
c     9 x 9 Gaussian filtering with "ifill".
c     -------------------------------------------------------------------
      do j=1,4
      do i=1,mszx146
              out_img1(i,j)=ifill
              out_img1wk(i,j)=ifill
      end do
      end do
      do j=mszy146-3,mszy146
      do i=1,mszx146
              out_img1(i,j)=ifill
              out_img1wk(i,j)=ifill
      end do
      end do
      do j=1,mszy146
      do i=1,4
              out_img1(i,j)=ifill
              out_img1wk(i,j)=ifill
      end do
      end do
      do j=1,mszy146
      do i=mszx146-3,mszx146
              out_img1(i,j)=ifill
              out_img1wk(i,j)=ifill
      end do
      end do
c     ------------------------------------------------------------
c     Segmentation of the current image by using ribcage edges.
c     Outside of ribcage edges are filled with "ifill".
c     ------------------------------------------------------------
      ibottomy=max(rribcage1(2,rribcage_no1)/istep,
     &             lribcage1(2,rribcage_no1)/istep)

      ixr=rribcage1(2,1)/istep
      if(ixr.eq.0) ixr=1
      do j=1,ixr
      do i=1,mszx146
              out_img1(i,j)=ifill
      end do
      end do

      do j=ibottomy,mszy146
      do i=1,mszx146
```

```
              out_img1(i,j)=ifill
      end do
      end do

      do i=1,rribcage_no1
              ixr=rribcage1(1,i)/istep
              if(ixr.eq.0) ixr=1
              do j=1, ixr
                      iry=rribcage1(2,i)/istep
                      if(iry.eq.0) iry=1
                      out_img1(j,iry)=ifill
              end do
      end do

      do i=1,lribcage_no1
              ilx=lribcage1(1,i)/istep
              if (ilx.eq.0) ilx=1
              do j=ilx,mszx146
                      ily=lribcage1(2,i)/istep
                      if(ily.eq.0) ily=1
                      out_img1(j,ily)=ifill
              end do
      end do

      do j=1,mszy146
              do i=1,mszx146
                  out_img1c(i,j)=out_img1(i,j)
              end do
      end do
c     ----------------------------------------------------------
c     Expand the ribcage edge for current image when the
c     ribcage edge is mis-detected, i.e., if it's inside lung.
c     Scheme for the upper right and the bottom right.
c     ----------------------------------------------------------
      ithd=10         ! threshold level for the edge.
      ilength=20      ! if the number of points of continuous line
                      ! is longer than "ilength", the line is expanded to
                      ! the outside from the mis-detected ribcage edge.
      idist=10        ! Maximum expanding range; 10 pixels
      ithd2=0         ! threshold level for stopping the edge search
!     ----------------------------------------------------------
!     Bottom right    (expansion to the right)
!     ----------------------------------------------------------
      ict=0
      do i=rribcage_no1,100,-1
         ixr=rribcage1(1,i)/istep
         iyr=rribcage1(2,i)/istep
         if(ixr.eq.0) ixr=1
         if(iyr.eq.0) iyr=1
         iedge=ifill-out_img1wk(ixr,iyr)
         if(iedge.gt.ithd) then
            ict=ict+1
         else
            ist=i+1
            iend=i+ict
            if (ict.ge.ilength) then
               do k=ist,iend
                  ixr2=rribcage1(1,k)/istep
                  iyr2=rribcage1(2,k)/istep
                  if(ixr2.eq.0) ixr2=1
                  if(iyr2.eq.0) iyr2=1
                  do l=0,idist
                     if(ixr2-l-1.le.0) goto 500
```

```
                     ie1=out_img1wk(ixr2-l,iyr2)
                     ie2=out_img1wk(ixr2-l-1,iyr2)
                     ie=ie2-ie1
                     if (ie.lt.ithd2) goto 500
                     out_img1c(ixr2-l,iyr2)=out_img1wk(ixr2-l,iyr2)
                  end do
500               continue
               end do
            end if
            ict=0
         end if
      end do
!     ------------------------------------------------------------
!     Upper right     (expansion to the top)
!     ------------------------------------------------------------
      ithd3=200      ! threshold level to avoid the clavicles trapping

      ict=0
      do i=99,1,-1
         ixr=rribcage1(1,i)/istep
         iyr=rribcage1(2,i)/istep
         if(ixr.eq.0) ixr=1
         if(iyr.eq.0) iyr=1
         iedge=ifill-out_img1wk(ixr,iyr)
         if(iedge.gt.ithd) then
            ict=ict+1
         else
            ist=i+1
            iend=i+ict
            if (ict.ge.ilength) then
               do k=ist,iend
                  ixr2=rribcage1(1,k)/istep
                  iyr2=rribcage1(2,k)/istep
                  if(ixr2.eq.0) ixr2=1
                  if(iyr2.eq.0) iyr2=1
                  do l=0,idist
                     if(iyr2-l-1.le.0) goto 511
                     ie1=out_img1wk(ixr2,iyr2-l)
                     ie2=out_img1wk(ixr2,iyr2-l-1)
                     ie3=ie2-ie1
                     if (ie3.gt.ithd2) then
                        out_img1c(ixr2,iyr2-l)=out_img1wk(ixr2,iyr2-l)
                     else
                        ie4=ifill-out_img1wk(ixr2,iyr2-l)
                        if (ie4.gt.ithd3.and.l.lt.2) then
                           out_img1c(ixr2,iyr2-l)=out_img1wk(ixr2,iyr2-l)
                        else
                           goto 511
                        end if
                     end if
                  end do
511               continue
               end do
            end if
            ict=0
         end if
      end do
!     ------------------------------------------------------------
!     Bottom left     (expansion to the left)
!     ------------------------------------------------------------
      ict=0
      do i=lribcage_no1,100,-1
         ixl=lribcage1(1,i)/istep
         iyl=lribcage1(2,i)/istep
         if(ixl.eq.0) ixl=1
         if(iyl.eq.0) iyl=1
         iedge=ifill-out_img1wk(ixl,iyl)
         if(iedge.gt.ithd) then
            ict=ict+1
         else
            ist=i+1
            iend=i+ict
            if (ict.ge.ilength) then
               do k=ist,iend
                  ixl2=lribcage1(1,k)/istep
                  iyl2=lribcage1(2,k)/istep
                  if(ixl2.eq.0) ixl2=1
                  if(iyl2.eq.0) iyl2=1
                  do l=0,idist
                     if(ixl2+l+1.ge.mszx146) goto 502
                     ie1=out_img1wk(ixl2+l,iyl2)
                     ie2=out_img1wk(ixl2+l+1,iyl2)
                     ie=ie2-ie1
                     if (ie.lt.ithd2) goto 502
                     out_img1c(ixl2+l,iyl2)=out_img1wk(ixl2+l,iyl2)
                  end do
502               continue
               end do
            end if
            ict=0
         end if
      end do
!     -----------------------------------------------------------
!     Upper left      (expansion to the top)
!     -----------------------------------------------------------
      ict=0
      do i=99,1,-1
         ixl=lribcage1(1,i)/istep
         iyl=lribcage1(2,i)/istep
         if(ixl.eq.0) ixl=1
         if(iyl.eq.0) iyl=1
         iedge=ifill-out_img1wk(ixl,iyl)
         if(iedge.gt.ithd) then
            ict=ict+1
         else
            ist=i+1
            iend=i+ict
            if (ict.ge.ilength) then
               do k=ist,iend
                  ixl2=lribcage1(1,k)/istep
                  iyl2=lribcage1(2,k)/istep
                  if(ixl2.eq.0) ixl2=1
                  if(iyl2.eq.0) iyl2=1
                  do l=0,idist
                     if(iyl2-l-1.le.0) goto 513
                     ie1=out_img1wk(ixl2,iyl2-l)
                     ie2=out_img1wk(ixl2,iyl2-l-1)
                     ie3=ie2-ie1
                     if (ie3.gt.ithd2) then
                        out_img1c(ixl2,iyl2-l)=out_img1wk(ixl2,iyl2-l)
                     else
                        ie4=ifill-out_img1wk(ixl2,iyl2-l)
                        if (ie4.gt.ithd3.and.l.lt.2) then
                           out_img1c(ixl2,iyl2-l)=out_img1wk(ixl2,iyl2-l)
                        else
                           goto 513
```

```
                        end if
                     end if
                  end do
513               continue
               end do
            end if
            ict=0
         end if
      end do

c     ----------------------------------------------------------------
c     Fill narrow space, if it exists.
c
c     Sometimes, the interval between X-locations of ribcage edges
c     is not 1, it may be appeared a line after the expansion process.
c     In order to avoid this, the line will be replaced with
c     original blurred low-resolution image data.
c     ----------------------------------------------------------------
      do j=1,mszy146
      do i=1,mszx146
              if(out_img1c(i,j).eq.ifill.and.out_img1c(i-1,j).lt.ifill.
     &           and.out_img1c(i+1,j).lt.ifill) then
                   out_img1c(i,j)=out_img1wk(i,j)
              end if
              if(out_img1c(i-1,j).eq.ifill.and.out_img1c(i+1,j).eq.ifill.
     &           and.out_img1c(i,j).lt.ifill) then
                   out_img1c(i,j)=ifill
              end if
              if(out_img1c(i,j).eq.ifill.and.out_img1c(i+1,j).eq.ifill.
     &           and.out_img1c(i-1,j).lt.ifill.and.
     &           out_img1c(i+2,j).lt.ifill) then
                   out_img1c(i,j)=out_img1wk(i,j)
                   out_img1c(i+1,j)=out_img1wk(i+1,j)
              end if
      end do
      end do
c     ----------------------------------------------------------------
c     Flip gray scale for cross-correlation
c     ----------------------------------------------------------------
      do j=1,mszy146
      do i=1,mszx146
              out_img1c(i,j)= 1023-out_img1c(i,j)
      end do
      end do
c     --------------------------------------------------------------------
c     Average along the ribcage edge line(image #2: previous image)
c     and determine the Max. and Min. values in the Mid. quadrant
c     area for previous image.
c     Average value along the ribcage edge is used for filling into
c     outside of ribcage edges.
c     Max. and Min. values in the Mid. quadrant area for previous image
c     are used for normalization of the blurred low-resolution images.
c
c     This part is related to the normalization of image.
c     --------------------------------------------------------------------
      avg2=0
      do i=1,rribcage_no2
              ixr=rribcage2(1,i)/istep
              iyr=rribcage2(2,i)/istep
              if(ixr.eq.0) ixr=1
              if(iyr.eq.0) iyr=1
              avg2=avg2+out_img2(ixr,iyr)
      end do
```

```
      do i=1,lribcage_no2
              ixl=lribcage2(1,i)/istep
              iyl=lribcage2(2,i)/istep
              if(ixl.eq.0) ilr=1
              if(iyl.eq.0) iyl=1
              avg2=avg2+out_img2(ixl,iyl)
      end do

      avg2=avg2/float(rribcage_no2+lribcage_no2) ! Average along the ribcage edge
!     ----------------------------------------------------------------
!     If the ribcage edges are mis-detected in the lung area, average
!     pixel value along the ribcage edge could be very low value.
!     To aboid this, minimum value of the avg2 is setted to 500.
!
c     This part is related to the normalization of image.
!     ----------------------------------------------------------------
      if (avg2.lt.500) avg2=500

c     write(*,*) 'Avg. on ribcage edge of previous image:',avg2

      max2=-1000                ! Maximum pixel value in mid. quadrant area
      min2= 1000                ! Minimun pixel value in mid. quadrant area
      do j=(mszy146/istep)*1,(mszy146/istep)*3
      do i=(mszx146/istep)*1,(mszx146/istep)*3
              if (out_img2(i,j).gt.max2) max2=out_img2(i,j)
              if (out_img2(i,j).lt.min2) min2=out_img2(i,j)
      end do
      end do

c     write(*,*) 'max2,min2',max2,min2

      ifil2=int(avg2)                 ! "ifil1" is the value for filling into
                                      ! outside of ribcage border.
c     -------------------------------------------------------------
c     Normalize for image #2: previous image)
c     -------------------------------------------------------------
      do j=1,mszy146
      do i=1,mszx146
              out_img2(i,j)= (1023/float((ifil2-min2)))*out_img2(i,j)
     &        -(1023*min2)/float((ifil2-min2))
              if(out_img2(i,j).lt.0) out_img2(i,j)=0
              if(out_img2(i,j).gt.1023) out_img2(i,j)=1023
              out_img2wk(i,j)= (1023/float((ifil2-min2)))*out_img2wk(i,j)
     &        -(1023*min2)/float((ifil2-min2))
              if(out_img2wk(i,j).lt.0) out_img2wk(i,j)=0
              if(out_img2wk(i,j).gt.1023) out_img2wk(i,j)=1023
      end do
      end do

      ifil2=1023                      ! After the low-resolution image
                                      ! is normalized, "ifill" is
                                      ! setted to 1023.
c     -------------------------------------------------------------------
c     Fill rims of the previous blurred low-resolution image caused by
c     9 x 9 Gaussian filtering with "ifil2".
c     -------------------------------------------------------------------
      do j=1,4
      do i=1,mszx146
              out_img2(i,j)=ifil2
              out_img2wk(i,j)=ifil2
      end do
      end do
      do j=mszy146-3,mszy146
```

```
      do i=1,mszx146
              out_img2(i,j)=ifil2
              out_img2wk(i,j)=ifil2
      end do
      end do
      do j=1,mszy146
      do i=1,4
              out_img2(i,j)=ifil2
              out_img2wk(i,j)=ifil2
      end do
      end do
      do j=1,mszy146
      do i=mszx146-3,mszx146
              out_img2(i,j)=ifil2
              out_img2wk(i,j)=ifil2
      end do
      end do
c     ---------------------------------------------------------------
c     Segmentation of the previous image by using ribcage edges.
c     Outside of ribcage edges are filled with "ifil2".
c     ---------------------------------------------------------------
      ibottomy=max(rribcage2(2,rribcage_no2)/istep,
     &             lribcage2(2,rribcage_no2)/istep)

      iry=rribcage2(2,1)/istep
      if(iry.eq.0) iry=1
      do j=1,iry
      do i=1,mszx146
              out_img2(i,j)=ifil2
      end do
      end do

      do j=ibottomy,mszy146
      do i=1,mszx146
              out_img2(i,j)=ifil1
      end do
      end do

      do i=1,rribcage_no2
              irx=rribcage2(1,i)/istep
              if(irx.eq.0) irx=1
              do j=1, irx
                      iry=rribcage2(2,i)/istep
                      if (iry.eq.0) iry=1
                      out_img2(j,iry)=ifil1
              end do
      end do

      do i=1,lribcage_no2
              ilx=lribcage2(1,i)/istep
              if (ilx.eq.0) ilx=1
              do j=ilx,mszx146
                      ily=lribcage2(2,i)/istep
                      if(ily.eq.0) ily=1
                      out_img2(j,ily)=ifil1
              end do
      end do

      do j=1,mszy146
      do i=1,mszx146
              out_img2c(i,j)=out_img2(i,j)
      end do
      end do
```

```
c       ---------------------------------------------------------
c       Expand the ribcage edge for previous image when the
c       ribcage edge is mis-detected, i.e., if it's inside lung.
c       Scheme for the upper right and the bottom right.
c       ---------------------------------------------------------
!       ---------------------------------------------------------
!       Bottom right.  (expansion to the right)
!       ---------------------------------------------------------
        ict=0
        do i=rribcage_no2,100,-1
           ixr=rribcage2(1,i)/istep
           iyr=rribcage2(2,i)/istep
           if(ixr.eq.0) ixr=1
           if(iyr.eq.0) iyr=1
           iedge=ifil2-out_img2wk(ixr,iyr)
           if(iedge.gt.ithd) then
              ict=ict+1
           else
              ist=i+1
              iend=i+ict
              if (ict.ge.ilength) then
                 do k=ist,iend
                    ixr2=rribcage2(1,k)/istep
                    iyr2=rribcage2(2,k)/istep
                    if(ixr2.eq.0) ixr2=1
                    if(iyr2.eq.0) iyr2=1
                    do l=0,idist
                       if(ixr2-l-1.le.0) goto 600
                       ie1=out_img2wk(ixr2-l,iyr2)
                       ie2=out_img2wk(ixr2-l-1,iyr2)
                       ie3=ie2-ie1
                       if (ie3.lt.ithd2) goto 600
                       out_img2c(ixr2-l,iyr2)=out_img2wk(ixr2-l,iyr2)
                    end do
600                 continue
                 end do
              end if
              ict=0
           end if
        end do
!       ---------------------------------------------------------
!       Upper right (expansion to the top)
!       ---------------------------------------------------------
        ict=0
        do i=99,1,-1
           ixr=rribcage2(1,i)/istep
           iyr=rribcage2(2,i)/istep
           if(ixr.eq.0) ixr=1
           if(iyr.eq.0) iyr=1
           iedge=ifil2-out_img2wk(ixr,iyr)
           if(iedge.gt.ithd) then
              ict=ict+1
           else
              ist=i+1
              iend=i+ict
              if (ict.ge.ilength) then
                 do k=ist,iend
                    ixr2=rribcage2(1,k)/istep
                    iyr2=rribcage2(2,k)/istep
                    if(ixr2.eq.0) ixr2=1
                    if(iyr2.eq.0) iyr2=1
                    do l=0,idist
                       if(iyr2-l-1.le.0) goto 611
```

```
                       ie1=out_img2wk(ixr2,iyr2-l)
                       ie2=out_img2wk(ixr2,iyr2-l-1)
                       ie3=ie2-ie1
                       if (ie3.gt.ithd2) then
                          out_img2c(ixr2,iyr2-l)=out_img2wk(ixr2,iyr2-l)
                       else
                          ie4=ifil2-out_img2wk(ixr2,iyr2-l)
                          if (ie4.gt.ithd3.and.l.lt.2) then
                             out_img2c(ixr2,iyr2-l)=out_img2wk(ixr2,iyr2-l)
                          else
                             goto 611
                          end if
                       end if
                    end do
611                 continue
                 end do
              end if
              ict=0
           end if
        end do
!       ---------------------------------------------------------
!       Bottom left  (expansion to the left)
!       ---------------------------------------------------------
        ict=0
        do i=lribcage_no2,100,-1
           ixl=lribcage2(1,i)/istep
           iyl=lribcage2(2,i)/istep
           if(ixl.eq.0) ixl=1
           if(iyl.eq.0) iyl=1
           iedge=ifil2-out_img2wk(ixl,iyl)
           if(iedge.gt.ithd) then
              ict=ict+1
           else
              ist=i+1
              iend=i+ict
              if (ict.ge.ilength) then
                 if(ixl2.eq.0) ixl2=1
                 if(iyl2.eq.0) iyl2=1
                 do k=ist,iend
                    ixl2=lribcage2(1,k)/istep
                    iyl2=lribcage2(2,k)/istep
                    do l=0,idist
                       if(ixl2+l+1.ge.mszx146) goto 602
                       ie1=out_img2wk(ixl2+l,iyl2)
                       ie2=out_img2wk(ixl2+l+1,iyl2)
                       ie3=ie2-ie1
                       if (ie3.lt.ithd2) goto 602
                       out_img2c(ixl2-l,iyl2)=out_img2wk(ixl2-l,iyl2)
                    end do
602                 continue
                 end do
              end if
              ict=0
           end if
        end do
!       ---------------------------------------------------------
!       Upper left      (expansion to the top)
!       ---------------------------------------------------------
        ict=0
        do i=99,1,-1
           ixl=lribcage2(1,i)/istep
           iyl=lribcage2(2,i)/istep
           if(ixl.eq.0) ixl=1
```

```
           if(iyl.eq.0) iyl=1
           iedge=ifil2-out_img2wk(ixl,iyl)
           if(iedge.gt.ithd) then
              ict=ict+1
           else
              ist=i+1
              iend=i+ict
              if (ict.ge.ilength) then
                 do k=ist,iend
                    ixl2=lribcage2(1,k)/istep
                    iyl2=lribcage2(2,k)/istep
                    if(ixl2.eq.0) ixl2=1
                    if(iyl2.eq.0) iyl2=1
                    do l=0,idist
                       if(iyl2-l-1.le.0) goto 613
                       ie1=out_img2wk(ixl2,iyl2-l)
                       ie2=out_img2wk(ixl2,iyl2-l-1)
                       ie3=ie2-ie1
                       if (ie3.gt.ithd2) then
                          out_img2c(ixl2,iyl2-l)=out_img2wk(ixl2,iyl2-l)
                       else
                          ie4=ifil2-out_img2wk(ixl2,iyl2-l)
                          if (ie4.gt.ithd3.and.l.lt.2) then
                             out_img2c(ixl2,iyl2-l)=out_img2wk(ixl2,iyl2-l)
                          else
                             goto 613
                          end if
                       end if
                    end do
613                 continue
                 end do
              end if
              ict=0
           end if
        end do
c       ---------------------------------------------------------
c       Fill narrow space, if it exists.
c
c       Sometimes, the interval between X-locations of ribcage edges
c       is not 1, it may be appeared a line after the expansion process.
c       In order to avoid this, the line will be replaced with
c       original blurred low-resolution image data.
c       ---------------------------------------------------------
        do j=1,mszy146
        do i=1,mszx146
                if(out_img2c(i,j).eq.ifil2.and.out_img2c(i-1,j).lt.ifil2.
     &          and.out_img2c(i+1,j).lt.ifil2) then
                        out_img2c(i,j)=out_img2wk(i,j)
                end if
                if(out_img2c(i-1,j).eq.ifil2.and.out_img2c(i+1,j).eq.ifil2.
     &          and.out_img2c(i,j).lt.ifil2) then
                        out_img2c(i,j)=ifil2
                end if
                if(out_img2c(i,j).eq.ifil2.and.out_img2c(i+1,j).eq.ifil2.
     &          and.out_img2c(i-1,j).lt.ifil2.and.
     &          out_img2c(i+2,j).lt.ifil2) then
                        out_img2c(i,j)=out_img2wk(i,j)
                        out_img2c(i+1,j)=out_img2wk(i+1,j)
                end if
        end do
        end do
c       ---------------------------------------------------------
c       Flip gray scales for cross-correlation
```

```
c       ---------------------------------------------------------
        do j=1,mszy146
            do i=1,mszx146
                out_img2c(i,j)= 1023-out_img2c(i,j)
            end do
        end do
c       ---------------------------------------------------------
c       Make template image from previous blurred low-resolution
c       image for cross-correlation.  (Matrix size : 100 x 60)
c       ---------------------------------------------------------
        ix2base=ribfeature2(6)/istep
        iy2base=min(rribcage2(2,1),lribcage2(2,1))/istep
        if(iy2base.eq.0) iy2base=1
        imarginx=50
        imarginy=0
        if(iy2base-imarginy.le.0) imarginy=0
c       write(*,*) ix2base,iy2base,imarginy,mszytmp
        do j=1,mszytmp
        do i=1,mszxtmp
                temp(i,j)= out_img2c(i+ix2base-imarginx,
     &          j+iy2base-imarginy)
        end do
        end do
c       ------------------------------------------------
c       Cross-correlation
c       -----------------------------------------------
        isa(1)=1
        isa(2)=1
        isa(3)=mszx146
        isa(4)=120

c       call corrr(out_img1c,temp,mszxtmp,mszytmp,mszx146,
c     &             mszy146,isa,JX,JY,cmax,corr_mat)

!-------------------------------------
!  using fine scan correlation
!  modified by Qiang Li
!-------------------------------------
        call corrr1(out_img1c,temp,mszxtmp,mszytmp,mszx146,
     &             mszy146,isa,JX,JY,cmax)
c       ------------------------------------------------
c       Cross-correlation values is saved as 'corr_map.dat'
c       -----------------------------------------------
c       open(unit=1,file='corr_map.dat',form='formatted')
c       do j=1,30
c               write(1,800) (corr_mat(i,j),i=1,28)
c       end do
c800    format (28f6.3)
c       close(1)
c       ------------------------------------------------
c       write results
c       -----------------------------------------------
c       write(*,*) 'Location at Max. correlation value:',jx,jy
c       write(*,*) '(0,0) = ',(ix2base-imarginx),
c     & (iy2base-imarginy)
c       write(*,*) '(0,0) means cross point of toplung and midline'
c       write(*,*) 'X-Y shift value (on 146x146 image)',
c     & (JX+imarginx)-ix2base,(JY+imarginy)-iy2base
c       write(*,*) 'Max. of correlation value:',cmax
c       call writeimage(out_img1c,'1.img' ,146,146)
c       call writeimage(temp,'2.img' ,100,60)

c       ixshift=(JX+imarginx)-ix2base ! X-shift value on low-resolution image
```

```
c       iyshift=(JY+imarginy)-iy2base ! Y-shift value on low-resolution image

!----------------------------------------
!  modified by Qiang Li
!----------------------------------------

        ixshift=(JX-1+imarginx)-ix2base ! X-shift value on low-resolution image
        iyshift=(JY-1+imarginy)-iy2base ! Y-shift value on low-resolution image

        ixshift=ixshift*istep           ! X-shift value on 512 image
        iyshift=iyshift*istep           ! Y-shift value on 512 image

        ixshift = int(ixshift* float(mszx_orig)/float(mszx))
        iyshift = int(iyshift* float(mszx_orig)/float(mszx))

        return
        end
```

```
c       =============================================================
        subroutine inversed(a,b,np,n)
c           matrix inversion by elimination with partial pivoting
c          a    ; original matrix
c          b    ; inverse matrix
c       5/16/92  BY S.KATSURAGAWA
c       =============================================================
        implicit integer*4 (i-n)
        real*8 a(np,np),b(np,np),eps,del,amax,atmp,btmp,div,amult
c
        eps=0.0000001d00
c       -------------------------------------------------------------
c       CONSTRUCT IDENTITY MATRIX B(I,J)=I
c       -------------------------------------------------------------
        do 6 i=1,n
                do 5 j=1,n
                        if(i-j)4,3,4
3                       b(i,j)=1.0
                        goto 5
4                       b(i,j)=0.0
5               continue
6       continue
c       -------------------------------------------------------------
c       LOCATE MAXIMUM MAGNITUDE A(I,K) ON OR BELOW MAIN DIAGONAL
c       -------------------------------------------------------------
        del=1.0
        do 45 k=1,n
                if(k-n)12,30,30
12              imax=k
                amax=dabs(a(k,k))
                kp1=k+1
                do 20 i=kp1,n
                        if(amax-dabs(a(i,k)))15,20,20
15                      imax=i
                        amax=dabs(a(i,k))
20              continue
c               -----------------------------------------------------
c               INTERCHANGE ROWS IMAX AND K IF IMAX NOT EQUAL TO K
c               -----------------------------------------------------
                if(imax-k)25,30,25
25              do 29 j=1,n
                        atmp=a(imax,j)
                        a(imax,j)=a(k,j)
                        a(k,j)=atmp
                        btmp=b(imax,j)
                        b(imax,j)=b(k,j)
                        b(k,j)=btmp
29              continue
                del=-del
30              continue
c               -----------------------------------------------------
c               TEST FOR SINGULAR MATRIX
c               -----------------------------------------------------
                if(dabs(a(k,k))-eps)93,93,35
35              del=a(k,k)*del
c               -----------------------------------------------------
c               DIVIDE PIVOT ROW BY ITS MAIN DIAGONAL ELEMENT
c               -----------------------------------------------------
                div=a(k,k)
                do 38 j=1,n
                        a(k,j)=a(k,j)/div
                        b(k,j)=b(k,j)/div
38              continue
```

```
c               -----------------------------------------------------
c               REPLACE EACH ROW BY LINEAR COMBINATION WITH PIVOT ROW
c               -----------------------------------------------------
                do 43 i=1,n
                        amult=a(i,k)
                        if(i-k)39,43,39
39                      do 42 j=1,n
                                a(i,j)=a(i,j)-amult*a(k,j)
                                b(i,j)=b(i,j)-amult*b(k,j)
42                      continue
43              continue
45      continue
c
99      return
c93      write(*,113)k
93      continue
        goto 99
113     format(1h ,'**** singular matrix for k=',i2,' ****')
        end
```

```
C*************DATA FITTING PROGRAM ******************************
C        NAME: KOFITC.FOR (FITTED BY AN ORTHOGONAL polynomial)
C        PROGRAMED BY KEN OHARA    10/31/85
C        RANGE OF DEGREES IS 1 - 20
C        CALL KOFITC(X,F,NN,MA,CF,MOPT)
C        X (INPUT)    : ARRAY VARIABLES,X(NN),
C        F (INPUT)    : ARRAY VALUES OF F(X),F(NN)
C        NN (INPUT)   : NUMBER OF DATA <=1200
C        MA (INPUT)   : MAXIMUM ORDER OF THE POLYNOMIAL. <=20
C        CF (OUTPUT) : ARRAY COEFFICENT
C             F(X)=CF(1)+CF(2)*X+CF(3)*X**2+CF(4)*X**3  .....ETC.
C       MOPT (input/OUTPUT) : ORDER OF THE FITTED POLYNOMIAL
C     NAUT : 0 ;SPECIFED THE MOPT, 1 ;SETTING DEGREE BY AKAIKE'S CRITERION
C ******************************************************************
        SUBROUTINE KOFITC2(X,F,NN,MA,CF,MOPT,NAUT,ierr)
        IMPLICIT INTEGER*2 (I-N)
        real*4 X(NN),F(NN),W(1200),AL(21),AC(21)
        real*4 P(1200),P1(1200),P2(1200),BE(21),B(21),CF(21)
        REAL*8 W1,W2,S,T,WX,WF

        DO J=1,1200
          w(j)=0.0
          p(j)=0.0
          p1(j)=0.0
          p2(j)=0.0
        end do


c       write(*,*)'kofitc2: MA, NN = ',  MA, NN
        IF(MA.GT.20) GOTO 999
        N=NN
        MMOPT=MOPT
        DO 10 J=1,21
        be(j)=0.0
        b(j)=0.0
        cf(j)=0.0
        AC(J)=0.0
        AL(J)=0.0
 10     CONTINUE
C
        MI=0
        Z=0.0
        O=1.0
        MN=MI+1
        MX=MA+1
        IF(MX.Le.N) GOTO 120
999     write(6,*) 'ORDER AND NUMBER OF DATA ARE WRONG !'
        ierr=0
        RETURN

C
 120    W1=X(1)
        W2=W1
        T=F(1)
        U=T
C
        IL=0
c       write(*,*)'kofitc2: MA = ', MA, ' IL = ', IL
        DO 20 J=1,N
        IF(IL.EQ.0)    W(J)=O
        IF(X(J).LT.W1) W1=X(J)
        IF(X(J).GT.W2) W2=X(J)
        IF(F(J).LT.T)  T=F(J)
```

```
        IF(F(J).GT.U)  U=F(J)

 20     CONTINUE
C
C       IL=0
        XM=(W1+W2)*0.5
        RA=2.0/(W2-W1)
        FM=(U+T)*0.5

        IF(U .EQ. T) THEN
          CF(1)=U
          DO I=2,21
            CF(I)=0.0
          END DO

          RETURN
        END IF

        RB=2.0/(U-T)
        S=Z
        WX=S
        W1=WX
C
        DO 290 J=1,N
        X(J)=(X(J)-XM)*RA
        F(J)=(F(J)-FM)*RB

        W1=W1+W(J)
        S=S+W(J)*F(J)
        WX=WX+W(J)*X(J)
 290    CONTINUE
C
        B(1)=S/W1
        S=Z
        DO 380 J=1,N
        P(J)=F(J)-B(1)

        S=S+P(J)*P(J)*W(J)
        P1(J)=O
        P2(J)=Z
 380    CONTINUE
C
        if(s.le.0.0) then
        mo=0
        goto 780
        endif
        AC(1)=DLOG(S)*N
        MO=0
        IF(MA.LE.0) GOTO 780
        W2=O
C
        DO 480 I=1,MA
        IP=I+1
        AL(I)=WX/W1
        BE(I)=W1/W2
        W2=W1
        W1=Z
        WX=Z
        WF=Z
        DO 540 J=1,N
        T=(X(J)-AL(I))*P1(J)-BE(I)*P2(J)
        WF=WF+T*W(J)*F(J)
        P2(J)=P1(J)
```

```
        P1(J)=T
        T=T*T*W(J)
        W1=W1+T
        WX=WX+T*X(J)
 540    CONTINUE
C
        B(IP)=WF/W1
        S=Z
        DO 650 J=1,N
        P(J)=P(J)-B(IP)*P1(J)
        S=S+P(J)*P(J)*W(J)
 650    CONTINUE
        if(s.le.0) then
        mo=i-1
        goto 735
        endif
        AC(IP)=DLOG(S)*N+I+I
 480    CONTINUE
C
C
 735    S=AC(MN)
        MO=MN-1
        DO 730 I=MN,MX
        IF(AC(I).GE.S) GOTO 730
        MO=I-1
        S=AC(I)
 730    CONTINUE
 780    IF(NAUT.EQ.1) THEN
        MP=MO
        MOPT=MO
        write(6,*) 'OPTIMUM ORDER OR POLYNOMIAL ',MP
        ELSE
        MP=MOPT
        MO=MOPT
        ENDIF
        DO 790 J=1,N
        X(J)=X(J)/RA+XM
        F(J)=F(J)/RB+FM
 790    CONTINUE
C
C     *******CALCULATION OF COEFFICENTS***********
C
        MOP=MP+1
        DO 890 K=1,MOP
        CF(K)=Z
        P(K)=Z
        P2(K)=Z
 890    CONTINUE
C
        CF(1)=B(1)/RB+FM
        IF(MO.EQ.0) GOTO 1110
        P(1)=O
        DO 940  I=1,MO
        IP=I+1
        U=Z
        W1=O
        DO 980  K=1,IP
        T=P(K)
        P(K)=U-AL(I)*T-BE(I)*P2(K)
        U=T
        P2(K)=T
        T=P(K)*B(IP)*W1
        DO 1040 J=1,K
```

```
        CF(K-J+1)=CF(K-J+1)+T/RB
        T=-T*(K-J)*XM/J
1040    CONTINUE
        W1=W1*RA
 980    CONTINUE
 940    CONTINUE
        RETURN
 1110   write(6,*) 'FITTING IS WRONG, PLEASE TRY ANOTHER METHOD'
        STOP
        END
```

```
C*************DATA FITTING PROGRAM ******************************
C       NAME: KOFITC.FOR (FITTED BY AN ORTHOGONAL polynomial)
C       PROGRAMED BY KEN OHARA    10/31/85
C       RANGE OF DEGREES IS 1 - 20
C       CALL KOFITC(X,F,NN,MA,CF,MOPT,NAUT)
C       X (INPUT)    : ARRAY VARIABLES,X(NN),
C       F (INPUT)    : ARRAY VALUES OF F(X),F(NN)
C       NN (INPUT)   : NUMBER OF DATA <=1200
C       MA (INPUT)   : MAXIMUM ORDER OF THE POLYNOMIAL. <=20
C       CF (OUTPUT) : ARRAY COEFFICENT
C            F(X)=CF(1)+CF(2)*X+CF(3)*X**2+CF(4)*X**3  .....ETC.
C      MOPT (input/OUTPUT) : ORDER OF THE FITTED POLYNOMIAL
C    NAUT : 0 ;SPECIFED THE MOPT, 1 ;SETTING DEGREE BY AKAIKE'S CRITERION
C ********************************************************************
        SUBROUTINE KOFITC(X,F,NN,MA,CF,MOPT,NAUT)
        IMPLICIT INTEGER*2 (I-N)
        real*4 X(NN),F(NN),W(1200),AL(21),AC(21)
        real*4 P(1200),P1(1200),P2(1200),BE(21),B(21),CF(21)
        REAL*8 W1,W2,S,T,WX,WF


        DO J=1,1200
          w(j)=0.0
          p(j)=0.0
          p1(j)=0.0
          p2(j)=0.0
        end do

c       write(*,*)'KOFITC_PACK: MA, NN = ',  MA, NN
        IF(MA.GT.20) GOTO 999
        N=NN
        MMOPT=MOPT
        DO 10 J=1,21
        be(j)=0.0
        b(j)=0.0
        cf(j)=0.0
        AC(J)=0.0
        AL(J)=0.0
 10     CONTINUE
C
        MI=0
        Z=0.0
        O=1.0
        MN=MI+1
        MX=MA+1
c       write(6,*) 'MX,N=',MX,N

        IF(MX.Le.N) GOTO 120
c       write(6,*) 'MX=',mx

999     write (6,*) 'ORDER AND NUMBER OF DATA ARE WRONG !'
        RETURN

C
 120    W1=X(1)
        W2=W1
        T=F(1)
        U=T
C
        IL=0
c       write(6,*) 'KOFITC_PACK: MA = ', MA, ' IL = ', IL
        DO 20 J=1,N
        IF(IL.EQ.0)    W(J)=O
```

```
        IF(X(J).LT.W1) W1=X(J)
        IF(X(J).GT.W2) W2=X(J)
        IF(F(J).LT.T)  T=F(J)
        IF(F(J).GT.U)  U=F(J)

 20     CONTINUE
C
C       IL=0
        XM=(W1+W2)*0.5
        RA=2.0/(W2-W1)
        FM=(U+T)*0.5

        IF(U .EQ. T) THEN
          CF(1)=U
          DO I=2,21
            CF(I)=0.0
          END DO

          RETURN
        END IF

        RB=2.0/(U-T)
        S=Z
        WX=S
        W1=WX
C
        DO 290 J=1,N
        X(J)=(X(J)-XM)*RA
        F(J)=(F(J)-FM)*RB

        W1=W1+W(J)
        S=S+W(J)*F(J)
        WX=WX+W(J)*X(J)
 290    CONTINUE
C
        B(1)=S/W1
        S=Z
        DO 380 J=1,N
        P(J)=F(J)-B(1)

        S=S+P(J)*P(J)*W(J)
        P1(J)=O
        P2(J)=Z
 380    CONTINUE
C
        if(s.le.0.0) then
        mo=0
        goto 780
        endif
        AC(1)=DLOG(S)*N
        MO=0
        IF(MA.LE.0) GOTO 780
        W2=O
C
        DO 480 I=1,MA
        IP=I+1
        AL(I)=WX/W1
        BE(I)=W1/W2
        W2=W1
        W1=Z
        WX=Z
        WF=Z
        DO 540 J=1,N
```

```
        T=(X(J)-AL(I))*P1(J)-BE(I)*P2(J)
        WF=WF+T*W(J)*F(J)
        P2(J)=P1(J)
        P1(J)=T
        T=T*T*W(J)
        W1=W1+T
        WX=WX+T*X(J)
 540    CONTINUE
C
        B(IP)=WF/W1
        S=Z
        DO 650 J=1,N
        P(J)=P(J)-B(IP)*P1(J)
        S=S+P(J)*P(J)*W(J)
 650    CONTINUE
        if(s.le.0) then
        mo=i-1
        goto 735
        endif
        AC(IP)=DLOG(S)*N+I+I
 480    CONTINUE
C
C
 735    S=AC(MN)
        MO=MN-1
        DO 730 I=MN,MX
        IF(AC(I).GE.S) GOTO 730
        MO=I-1
        S=AC(I)
 730    CONTINUE
 780    IF(NAUT.EQ.1) THEN
        MP=MO
        MOPT=MO
        write (6,*) 'OPTIMUM ORDER OR POLYNOMIAL ',MP
        ELSE
        MP=MOPT
        MO=MOPT
        ENDIF
        DO 790 J=1,N
        X(J)=X(J)/RA+XM
        F(J)=F(J)/RB+FM
 790    CONTINUE
C
C    *******CALCULATION OF COEFFICENTS***********
C
        MOP=MP+1
        DO 890 K=1,MOP
        CF(K)=Z
        P(K)=Z
        P2(K)=Z
 890    CONTINUE
C
        CF(1)=B(1)/RB+FM
        IF(MO.EQ.0) GOTO 1110
        P(1)=O
        DO 940  I=1,MO
        IP=I+1
        U=Z
        W1=O
        DO 980  K=1,IP
        T=P(K)
        P(K)=U-AL(I)*T-BE(I)*P2(K)
        U=T
```

```
        P2(K)=T
        T=P(K)*B(IP)*W1
        DO 1040 J=1,K
        CF(K-J+1)=CF(K-J+1)+T/RB
        T=-T*(K-J)*XM/J
1040    CONTINUE
        W1=W1*RA
 980    CONTINUE
 940    CONTINUE
        RETURN
 1110   write (6,*) 'FITTING IS WRONG, PLEASE TRY ANOTHER METHOD'
        STOP
        END

C**********************************************************************
C
C        SUBROUTINE POLYFITC_integer(IY,CF,FITO,IX)
C
C        INPUT AND OUTPUT ARE INTEGER
C
C        IY: INPUT, AUTOVARIABLE VALUE
C        CF(21):INPUT, COEFFICIENT
C        FITO: INPUT, ACTUAL FIT ORDER
C        IX: OUTPUT,FITTED VALUE
C
C**********************************************************************
C
C
       SUBROUTINE POLYFITC_integer(IY,CF,FITO,IX)
C
C
       IMPLICIT INTEGER*2 (I-N)
       INTEGER*2 FITO
       REAL*4 CF(21)
       REAL*8 XXX,SUM
C
C
       Y=FLOAT(IY)
       MMM=FITO+1
       SUM=CF(1)
       DO 20 I=2,MMM
         XXX=CF(I)*(Y**(I-1))
         SUM=SUM+XXX
20     CONTINUE
       IX=INT(SUM)
C
       RETURN
       END
C**********************************************************************
C
C        SUBROUTINE POLYFITC_real(Y,CF,FITO,X)
C
C        INPUT AND OUTPUT ARE REAL VALUE
C
C        Y: INPUT, AUTOVARIABLE VALUE
C        CF(21):INPUT, COEFFICIENT
C        FITO: INPUT, ACTUAL FIT ORDER
C        X: OUTPUT,FITTED VALUE
C
C**********************************************************************
C
C
       SUBROUTINE POLYFITC_real(Y,CF,FITO,X)
```

```
C
C
      IMPLICIT INTEGER*2 (I-N)
      INTEGER*2 FITO
      REAL*4 CF(21)
      REAL*8 XXX,SUM
C
C
      MMM=FITO+1
      SUM=CF(1)
      DO 20 I=2,MMM
        XXX=CF(I)*(Y**(I-1))
        SUM=SUM+XXX
20    CONTINUE
      X=SUM
C
      RETURN
      END
      SUBROUTINE kyoukai(IP,JP,ISX,ISY,ISW)
C
CS    CALL kyoukai(IP,JP,ISX,ISY,ISW)
C
CP        2-CHI GAZOU NO KYOUKAI(EDGE) O KENSYUTSU SURU.
CP          (4-RENKETSU)
C
C
CA             IP(ISX,ISY) * NYURYOKU 2-CHI GAZOU     (IN)
CA             JP(ISX,ISY) * SYUTSURYOKU EDGE         (OUT)
CA             ISW         * SWITCH                   (IN)
CA                             IS.EQ.1 --- UCHIGAWA
CA                             IS.NE.1 --- SOTOGAWA
C
CN        SYUTSURYOKO EDGE WA 8-RENKETSU DE MOTOME RARERU.
C
C
CK    BORDER, BOUNDARY, BINARY IMAGE
C
CD                  1979.06.13     M.TEZUKA
C
C
      integer*2  IP(ISX,ISY)
      integer*2  JP(ISX,ISY)
C
C
      JSX = ISX
      JSY = ISY
C
      ID = 1
      IF (ISW .NE. 1) ID = 0
      IN = 1-ID
C
C
      DO 300 IY=1,JSY
      IYM1 = MAX0(1,IY-1)
      IYP1 = MIN0(JSY,IY+1)
C
      DO 300 IX=1,JSX
      JPD = 0
      IF (IP(IX,IY) .NE. ID) GO TO 200
C
      IXM1 = MAX0(1,IX-1)
      IXP1 = MIN0(JSX,IX+1)
C
      IF (IP(IX,IYM1) .EQ. IN) GO TO 100
      IF (IP(IXM1,IY) .EQ. IN) GO TO 100
      IF (IP(IXP1,IY) .EQ. IN) GO TO 100
      IF (IP(IX,IYP1) .NE. IN) GO TO 200
C
  100 CONTINUE
      JPD = 1
C
  200 CONTINUE
      JP(IX,IY) = JPD
C
  300 CONTINUE
C
C
      RETURN
      END
```

```
C------------------------------------------------------------------------
C
C   PROGRAM NAME:  LINE          (TYPE:  SUBROUTINE)
C   FUNCTION    :  Make WHITE BOLD LINE BETWEEN TWO POINTS
C
C------------------------------------------------------------------------
C
C               AUTHOR     DATE(DD-MMM-YY)
CD  CODED   BY  M.SONE          30-JUN-85
CD  REVISED BY  T.KOMATSU       25-FEB-86
C
C   PURPOSE:
C
CP      GENERATION OF LINES
CP
CP
C
C   CALLING SEQUENCE:
C
CS      CALL LINE3(KP,ISX,ISY,IX1,IY1,IX2,IY2,JERR)
C
C   ARGUMENT(S)   = TYPE  I/O      ;  COMMENT
C
CA    KP(ISX,ISY) =       I/O      ;  IMAGE ARRAY IN WHICH THE
CA                                 ;   GENERATED LINES ARE STORED
CA    IX1,IY1     =        I       ;  COORDINATE OF CURRENT POINT
CA    IX2,IY2     =        I       ;  COORDINATE OF NEXT POINT
CA    JERR        =        O       ;  JERR=0     NO ERROR
CA                                    JERR=-1    IX1.LE.0
CA                                               IY1.LE.0
CA                                               IX2.GT.ISX
CA                                               IY2.GT.ISY
C
C   SUBPROGRAMS USED IN THIS PROGRAM:
C
CL      NONE
C
C   NOTE:
C
CN      REFERECCE
CN      J.E.BRESENHAM,"ALGORITHM FOR COMPUTER CONTROL OF A DIGITAL
CN      PLOTTER",IBM SYSTEMS JOURNAL,VOL.4,NO.1,PP.25-30(1965).
C
C------------------------------------------------------------------------
C   KEYWORDS:
CK      LINE
C------------------------------------------------------------------------
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
```

```
C
      SUBROUTINE LINE3(KP,ISX,ISY,IX1,IY1,IX2,IY2,JERR)
C
C
      integer*2 KP(ISX,ISY)
C
      IC=IX2-IX1+IY2-IY1
      IF(IC.LT.0) GO TO 100
      GO TO 150
  100 IA=IX1
      IB=IY1
      IX1=IX2
      IY1=IY2
      IX2=IA
      IY2=IB
  150 CONTINUE
C
      IF(IX1.LE.0.OR.IX1.GT.ISX) GO TO 550
      IF(IX2.LE.0.OR.IX2.GT.ISX) GO TO 550
      IF(IY1.LE.0.OR.IY1.GT.ISY) GO TO 550
      IF(IY2.LE.0.OR.IY2.GT.ISY) GO TO 550
C
      JERR=0
      KP(IX1,IY1)=1023
      KP(IX1+1,IY1)=1023

      KP(IX1,IY1+1)=1023

      KP(IX1+1,IY1+1)=1023

      IDX=IABS(IX2-IX1)
      IDY=IABS(IY2-IY1)
C **********************************************
      IS=IX2-IX1
      IF(IC.EQ.0 .AND. IS.GT.0) GO TO 220
      IF(IDX.LE.IDY) GO TO 200
  220 CONTINUE
      ICON2=2*IDY
      IE=ICON2-IDX
      ICON1=IE-IDX
      IF(IY1.LT.IY2) INC=1
      IF(IY1.GE.IY2) INC=-1
      IY=IY1
      IZZ=IX1+1
      DO 300 IX=IZZ,IX2
          IF(IE.GT.0) GO TO 250
          IF(IE.LE.0) GO TO 350
C
CF    SELECT (IX+1,IY+INC)
C
  250     IY=IY+INC
          IE=IE+ICON1
          GO TO 270
C
CF    SELECT (IX+1,IY)
C
  350     IE=IE+ICON2
C
  270     KP(IX,IY)=1023
       KP(IX+1,IY)=1023
       KP(IX,IY+1)=1023
       KP(IX+1,IY+1)=1023
  300 CONTINUE
```

```
C
CF    RESET
C
      IF(IC.GE.0) GO TO 310
      IX2=IX1
      IY2=IY1
      IX1=IA
      IY1=IB
  310 CONTINUE
      GO TO 600
C ***********************************************
  200 CONTINUE
      ICON2=2*IDX
      IE=ICON2-IDY
      ICON1=IE-IDY
      IF(IX1.LT.IX2) INC=1
      IF(IX1.GE.IX2) INC=-1
      IX=IX1
      IZZ=IY1+1
      DO 400 IY=IZZ,IY2
          IF(IE.GT.0) GO TO 450
          IF(IE.LE.0) GO TO 500
C
CF    SELECT (IX+INC,IY+1)
C
  450     IX=IX+INC
          IE=IE+ICON1
          GO TO 470
C
CF    SELECT (IX,IY+1)
C
  500     IE=IE+ICON2
C
  470     KP(IX,IY)=1023
       KP(IX+1,IY)=1023
       KP(IX,IY+1)=1023
       KP(IX+1,IY+1)=1023
  400 CONTINUE
C
CF    RESET
C
      IF(IC.GE.0) GO TO 510
      IX2=IX1
      IY2=IY1
      IX1=IA
      IY1=IB
  510 CONTINUE
      GO TO 600
C ***********************************************
  550 CONTINUE
      JERR=-1
  600 RETURN
      END
```

```
!=========================================================================
      function Local_Matching_Skip( image1, image2, ncol, nlin, sas, tps,
      1                number, sac, tpc, region1, skip, DX, DY, CC )
!=========================================================================
!     Ver. 1.0
!     Written by Akiko Kano,  Apr.12, 1993
!-------------------------------------------------------------------------
!     This function performs a local matching for pairs of ROIs with reduced
!     matrix sizes, based on a cross-correlation method.
!     Normalized cross-correlation value CC is defined as the summation of
!     (a(i,j)-A)*(b(i,j)-B)/(sa*sb), for i=1,2,..,n and j=1,2,..,n, where
!     matrix a(i,j) is a template ROI, b(i,j) is a subregion in the corres-
!     ponding search area ROI, A and B is the average of a(i,j) and b(i,j),
!     respectively, and sa and sb is the variance of a(i,j) and b(i,j),
!     respectively.
!     This function searches the best match subregion which maximizes CC for
!     each template, by using a reduced matrix size with skipping pixels.
!     This function returns the CC and shift values DX and DY, which corres-
!     pond to the difference in x,y-location between the center of the best
!     match subregion and the center of the template.
!
!     (1) Both template size and search area size must be "skip*2N+1" (N:
!         integer), and distance increment must be "skip*M" (M:integer).
!     (2) Templates are selected on Image2 and search areas are selected on
!         Image1.
!     (3) Calculation of CC is based on a subroutine in "SPIDER".
!     (4) Search of subregions is based on a coarse-and-fine search method
!         in "SPIDER".
!     (5) Returns "error" if the average cross-correlation value is less than
!         0.5.
!-------------------------------------------------------------------------
!     This function calls;    CORR2
!=========================================================================
!     ARGUMENTS
!-------------------------------------------------------------------------
      implicit  none
      integer*4 MAXPT
      parameter (MAXPT=3000)         ! Maximum No. of ROI Pairs

      integer*4 Local_Matching_Skip
      integer*4 ncol, nlin           ! Matrix Size of Image Data          [I]
      integer*2 image1(ncol,nlin)    ! Original Image Data 1              [I]
      integer*2 image2(ncol,nlin)    ! Original Image Data 2              [I]
      integer*4 tps                  ! Template ROI Size (Pixels)         [I]
      integer*4 sas                  ! Search Area ROI Size (Pixels)      [I]
      integer*4 number               ! Number of ROI Pairs                [I]
      integer*4 sac(2,MAXPT)         ! Centers of Search Areas on Image1  [I]
      integer*4 tpc(2,MAXPT)         ! Centers of Templates on Image2     [I]
      integer*4 region1(4)           ! Smallest Rect. Area Including sac  [I]
      integer*4 skip                 ! Matrix Size Reduction Rate         [I]
      integer*4 DX(MAXPT), DY(MAXPT)         ! Shift Values
[O]
      real*4    CC(MAXPT)                    ! Cross-Correlation Values
[O]
      real*4    LOW_CC

!-------------------------------------------------------------------------
!     VARIABLES
!-------------------------------------------------------------------------
c     parameter (LOW_CC=0.5)
      parameter (LOW_CC=0.2)
      integer*4 isa(4)
      integer*4 hsas, htps, stps
```

```
      integer*4 rlimit, llimit, tlimit, blimit
      integer*4 SC1, SL1, EC1, EL1, CC1, CL1
      integer*4 SC2, SL2, EC2, EL2
      integer*4 ncs, nls
      integer*4 JX, JY, N

      integer*4 MAXDIM
      parameter (MAXDIM=1000)
      integer*4 buf_sa_skip(MAXDIM,MAXDIM), buf_tp_skip(MAXDIM,MAXDIM)  ! Shige

      real*4    ave_ccv
!-------------------------------------------------------------------
!     Pointers
!-------------------------------------------------------------------
!     pointer (p_sa_skip, buf_sa_skip), (p_tp_skip, buf_tp_skip)

!-------------------------------------------------------------------------
!     FUNCTIONS
!-------------------------------------------------------------------------
c     integer*4 Convert_for_Spider_Skip

c     integer*4 PutOutputF
      real*8    Sprint
      integer   malloc
!-------------------------------------------------------------------------
♀

!-------------------------------------------------------------------------
!     1. DEFINE HALF ROI SIZES
!-------------------------------------------------------------------------
      hsas = sas / 2
      htps = tps / 2
!-------------------------------------------------------------------------
!     2. DETERMINE THE REGION INCLUDING WHOLE SEARCH AREAS
!-------------------------------------------------------------------------
      rlimit = region1(1) - skip * ( (region1(1)-1) / skip )
      tlimit = region1(2) - skip * ( (region1(2)-1) / skip )
      llimit = region1(3) + skip * ( (ncol-region1(3)) / skip )
      blimit = region1(4) + skip * ( (nlin-region1(4)) / skip )

      SC1 = max( region1(1)-hsas, rlimit )
      SL1 = max( region1(2)-hsas, tlimit )
      EC1 = min( region1(3)+hsas, llimit )
      EL1 = min( region1(4)+hsas, blimit )
      ncs = ( EC1 - SC1 ) / skip + 1
      nls = ( EL1 - SL1 ) / skip + 1
!-------------------------------------------------------------------------
!     3. GET BUFFER MEMORY FOR THE SEARCH REGION WITH REDUCED MATRIX SIZE
!-------------------------------------------------------------------------
c     call LIB$GET_VM( ncs*nls*4, buf_sa_skip )
!     p_sa_skip = malloc( ncs*nls*4 )
      if (ncs .gt. MAXDIM .or. nls .gt. MAXDIM) then
        write(*,*) 'Too Large Dimension of buf_sa_skip'
        stop
      end if
      call Convert_For_Spider_Skip( image1, buf_sa_skip, ncol, nlin,
      1                                      ncs, nls, SC1, SL1, skip )

c     call PutOutputF(' ')
c     write(*,*) ' '
      ave_ccv = 0.0

      stps = tps / skip + 1
```

```
c         call LIB$GET_VM( stps*stps*4, buf_tp_skip )
!     p_tp_skip = malloc( stps*stps*4 )
      if (stps .gt. MAXDIM) then
        write(*,*) 'Too Large Dimension of buf_tp_skip'
        stop
      end if

      do N = 1, number
!-------------------------------------------------------------------------
!         4. LOCATE SEARCH AREA WITH REDUCED MATRIX SIZE
!-------------------------------------------------------------------------
          CC1 = ( sac(1,N) - SC1 ) / skip + 1
          CL1 = ( sac(2,N) - SL1 ) / skip + 1

          isa(3) = min( CC1+hsas/skip, ncs )
          isa(4) = min( CL1+hsas/skip, nls )
          isa(1) = max( CC1-hsas/skip, 1 )
          isa(2) = max( CL1-hsas/skip, 1 )
!-------------------------------------------------------------------------
!         5. LOCATE TEMPLATE WITH REDUCED MATRIX SIZE
!-------------------------------------------------------------------------
          SC2 = tpc(1,N) - htps
          SL2 = tpc(2,N) - htps
!-------------------------------------------------------------------------
!         6. GET BUFFER MEMORY FOR THE TEMPLATE WITH REDUCED MATRIX SIZE
!-------------------------------------------------------------------------
          call Convert_For_Spider_Skip( image2, buf_tp_skip, ncol,nlin,
      1                                    stps, stps, SC2, SL2, skip )
!-------------------------------------------------------------------------
!         7. FIND THE BEST MATCH WITH REDUCED MATRIX SIZE
!-------------------------------------------------------------------------

!-------------------------------------
!  using fine scan correlation
!  modified by Qiang Li
!-------------------------------------
          call CORR1( buf_sa_skip, buf_tp_skip,
      1                          stps, stps, ncs, nls, isa, JX, JY, CC(N) )

c         DX(N) = 0
c         DY(N) = 0
          DX(N) = SC1 + ( JX - 1 ) * skip - SC2
          DY(N) = SL1 + ( JY - 1 ) * skip - SL2
c         write (*,*) N, JX, JY, DX(N), DY(N), CC(N)

          ave_ccv = ave_ccv + CC(N)

c         if ( mod(N,50).eq.0 ) write(*,*) '.:.', N, '/', ave_ccv/float(N)
c         if ( mod(N,300).eq.0 ) write(*,*) ' '
!-------------------------------------------------------------------------
!         8. CLEAR THE BUFFER
!-------------------------------------------------------------------------
c         call LIB$FREE_VM( stps*stps*4, buf_tp_skip )
      end do

!     call free( p_tp_skip )
c     call LIB$FREE_VM( ncs*nls*4, buf_sa_skip )
!     call free( p_sa_skip )
!-------------------------------------------------------------------------
!     9. CHECK THE AVERAGE CROSS-CORRELATION VALUE
!-------------------------------------------------------------------------
      ave_ccv = ave_ccv / float(number)
```

```
      if ( ave_ccv.lt.LOW_CC ) then
          write(*,*)
      1          '!!! Average cross-correlation value ', ave_ccv, ' is too small.
!!!'
          Local_Matching_Skip = 0
      else
          Local_Matching_Skip = 1
      end if
c     write(*,*) ' '
!-------------------------------------------------------------------------
!     END
!-------------------------------------------------------------------------
      return
      end
♀
!=========================================================================
      subroutine Convert_For_Spider_Skip( image, image_sp, ncol, nlin,
      1                                   rcol, rlin, scol, slin, skip )
!=========================================================================
!     Ver. 1.0
!     Written by Akiko Kano,  Apr.9, 1993
!-------------------------------------------------------------------------
!     This function converts a part of image data to an integer*4 array for
!     use in subroutines in "SPIDER".
!=========================================================================
!     ARGUMENTS
!-------------------------------------------------------------------------
      implicit  none
c     integer*4 Convert_for_Spider_Skip
      integer*4 ncol, nlin             ! Matrix Size of Original Image Data[I]
      integer*4 rcol, rlin             ! Matrix Size of Spider Image Data  [I]
      integer*4 scol, slin             ! Start Point                       [I]
      integer*2 image(ncol,nlin)       ! Original Image Data               [I]
      integer*4 image_sp(rcol,rlin)    ! Spider Image Data                 [O]
      integer*4 skip                   ! Reduction Rate of Matrix Size     [I]
!-------------------------------------------------------------------------
!     VARIABLES
!-------------------------------------------------------------------------
      integer*4 ecol, elin
      integer*4 C, L, CC, LL, data
!-------------------------------------------------------------------------
!     BEGIN
!-------------------------------------------------------------------------
      ecol = scol + skip * (rcol-1)
      elin = slin + skip * (rlin-1)
      if ( ecol.gt.ncol .or.elin.gt.nlin ) then
          write(*,*) '!!! Invalid ROI size ', rcol, ' X ', rlin, '.!!!'
c         Convert_for_Spider_Skip = 0
      else
          do L = slin, elin, skip
              do C = scol, ecol, skip
                  data = image(C,L)
                  CC = (C-scol) / skip + 1
                  LL = (L-slin) / skip + 1
                  image_sp(CC,LL) = data
              end do
          end do
c         Convert_for_Spider_Skip = 1
      end if
!-------------------------------------------------------------------------
!     END
!-------------------------------------------------------------------------
      return
```

```
end
```

```
c************************************************************************
c
c    subroutine: lung_boundary
c
c    This subroutine is to combine ribcage detection
c       and diaphragm detection together to delineate
c       lung boundary.
c
c    Input: image(nxw,nyh), image buffer: 1k by 1k
c    Output: R,Lribcage; R,Ldiaph: ribcage & diaphragm coordinates
c
c    Date: 12/9/93  V1
c
c       link with subroutines: ribcage_detection,ribcagepoint3
c       chest_pack3,kofitc_pack,diaphragm_detection
c
c
c       Xin-Wei Xu
c
c************************************************************************

        subroutine lung_boundary(image,ncol,nlin,feature,
     %          Rribcage,Rribcage_No,
     %          Lribcage,Lribcage_No,
     %          Rdiaph,Rdiaph_No,
     %          Ldiaph,Ldiaph_No,
     %          Rindex,Lindex,cf_rib_r,cf_rib_l,cf_dia_r,cf_dia_l)

!========================================================================
!       ARGUMENTS
!========================================================================

        implicit integer*2 (i-n)

        integer*4 ncol,nlin
        integer*2 image(ncol,nlin) ! image buffer

        integer*2
     &            Rribcage(2,1215),Rribcage_No, ! R ribcage output
     &            Lribcage(2,1215),Lribcage_No, ! L ribcage output

     %            Rdiaph(2,1215),Rdiaph_No,  ! R diaphragm output
     %            Ldiaph(2,1215),Ldiaph_No   ! L diaphragm output

        integer*2 feature(50) ! landmarks of chest, from ribcage detection

        integer*2 Rindex ! =1 right tip angle close; =2 not;
        integer*2 Lindex ! =1 left tip angle close; =2 not;
        real*4          cf_rib_r(21),cf_rib_l(21)
        real*4          cf_dia_r(21),cf_dia_l(21)

!========================================================================
!       VARIABLES
!========================================================================

        integer*2 temp(2,1215)

!========================================================================

c************************ Begin *************************************

        Rindex=1
        Lindex=1
```

```
        call ribcage_detection(Image,ncol,nlin,feature,
     &                  Rribcage,Rribcage_No,
     &          Lribcage,Lribcage_No,cf_rib_r,cf_rib_l)

cc      write(6,*)' Finish ribcage detection '
cc      write(6,*)'Rribcage:',Rribcage_No,Rribcage(1,Rribcage_No),
cc     %Rribcage(2,Rribcage_No)

cc      write(6,*)'Lribcage:',Lribcage_No,Lribcage(1,Lribcage_No),
cc     %Lribcage(2,Lribcage_No)

!************************************************************************

        call diaphragm_detection(image,ncol,nlin,feature,
     %                  Rdiaph,Rdiaph_No,
     %                  Ldiaph,Ldiaph_No,cf_dia_r,cf_dia_l)

cc      write(6,*)' Finish diaphragm detection '
cc      write(6,*)'Rdiaph:',Rdiaph_No,Rdiaph(1,1),
cc     %Rdiaph(2,1)

cc      write(6,*)'Ldiaph:',Ldiaph_No,Ldiaph(1,Ldiaph_No),
cc     %Ldiaph(2,Ldiaph_No)

c ***********************************************************************

        do i=1,1215
          temp(1,i)=0
          temp(2,i)=0
        end do

        if (Rdiaph(2,1).gt.nlin-1) then

          Rindex=2
          do i=1,Rdiaph_No
            if (Rdiaph(2,i).le.nlin-1) then
              jstart=i
              go to 10
            end if
          end do
10        continue
          j=0
          do i=jstart,Rdiaph_No
            j=j+1
            temp(1,j)=Rdiaph(1,i)
            temp(2,j)=Rdiaph(2,i)
          end do
          Rdiaph_No=j

          do i=1,1215
            Rdiaph(1,i)=0
            Rdiaph(2,i)=0
          end do

          do i=1,Rdiaph_No
            Rdiaph(1,i)=temp(1,i)
            Rdiaph(2,i)=temp(2,i)
          end do

        end if

        if (Rribcage(2,Rribcage_No).le.Rdiaph(2,1)) then
```

```
          NR=Rdiaph(2,1)-Rribcage(2,Rribcage_No)
          iy0=Rribcage(2,Rribcage_No)
          ix0=Rribcage(1,Rribcage_No)
          do j=1,NR
            Rribcage_No=Rribcage_No+1
            Rribcage(2,Rribcage_No)=iy0+j
            Rribcage(1,Rribcage_No)=ix0
          end do
        end if
cc      write(6,*)'NR=',NR

        if (Rindex.eq.2) then
          do i=1,Rdiaph_No
            temp(1,i)=Rdiaph(1,i)
            temp(2,i)=Rdiaph(2,i)
          end do
          j=0
          do i=Rribcage(1,Rribcage_No),Rdiaph(1,1)
            j=j+1
            Rdiaph(1,j)=i
            Rdiaph(2,j)=nlin-1
          end do
          do i=1,Rdiaph_No
            k=j+i
            Rdiaph(1,k)=temp(1,i)
            Rdiaph(2,k)=temp(2,i)
          end do
          Rdiaph_No=k
        end if
!------------------------------------------------------------------------

        do i=1,1215
          temp(1,i)=0
          temp(2,i)=0
        end do

        if (Ldiaph(2,Ldiaph_No).gt.nlin-1) then

          Lindex=2
          do i=Ldiaph_No,1,-1
            if (Ldiaph(2,i).le.nlin-1) then
              jend=i
              go to 20
            end if
          end do
20        continue
          j=0
          do i=1,jend
            j=j+1
            temp(1,j)=Ldiaph(1,i)
            temp(2,j)=Ldiaph(2,i)
          end do
          Ldiaph_No=j

          do i=1,1215
            Ldiaph(1,i)=0
            Ldiaph(2,i)=0
          end do

          do i=1,Ldiaph_No
            Ldiaph(1,i)=temp(1,i)
            Ldiaph(2,i)=temp(2,i)
          end do
```

```
        end if

        if (Lribcage(2,Lribcage_No).le.Ldiaph(2,Ldiaph_No)) then
          NL=Ldiaph(2,Ldiaph_No)-Lribcage(2,Lribcage_No)
          iy0=Lribcage(2,Lribcage_No)
          ix0=Lribcage(1,Lribcage_No)
          do j=1,NL
            Lribcage_No=Lribcage_No+1
            Lribcage(2,Lribcage_No)=iy0+j
            Lribcage(1,Lribcage_No)=ix0
          end do
        end if
cc      write(6,*)'NL=',NL

        if (Lindex.eq.2) then
          j=0
          do i=Ldiaph(1,Ldiaph_No),Lribcage(1,Lribcage_No)
            j=j+1
            temp(1,j)=i
            temp(2,j)=nlin-1
          end do
          do i=1,j
            k=Ldiaph_No+i
            Ldiaph(1,k)=temp(1,i)
            Ldiaph(2,k)=temp(2,i)
          end do
          Ldiaph_No=Ldiaph_No+j
        end if

****************************** end ***************************************

        return
        end
```

```
/*=======================================================================
===
      Subroutine lut_filename_
      Determination of Filename of Nonlinear Density Correction LUT
      Coded by Shige
========================================================================
===*/
#include    <stdio.h>
#include        <string.h>
#include    <stdlib.h>
#include    <time.h>

#ifdef KRL_GNU
  #define lut_filename lut_filename__
#else
  #define lut_filename lut_filename_
#endif

extern "C" void lut_filename(char *dirlut, char *head, char *density,
                        char *tableNo, char* tail, char *filename)
{
sprintf(filename, "%s%s%s%s%s", dirlut,
                        head,
                        density,
                        tableNo,
                        tail);
}
```

```
/*======================================================================
===
      Subroutine lut_filename_
      Determination of Filename of Nonlinear Density Correction LUT
      Coded by Shige
========================================================================
===*/
#include    <stdio.h>
#include        <string.h>
#include    <stdlib.h>
#include    <time.h>

#ifdef KRL_GNU
  #define lut_filename lut_filename__
#else
  #define lut_filename lut_filename_
#endif

extern "C" void lut_filename(char *dirlut, char *head, char *density,
                        char *tableNo, char* tail, char *filename)
{
sprintf(filename, "%s%s%s%s%s", dirlut,
                        head,
                        density,
                        tableNo,
                        tail);
}
```

```
/*=====================================================================
===
      Subroutine nonlinear_density_correction_
      Nonlinear Density Correction
      Coded by Shige
=======================================================================
===*/
#include    <stdio.h>
#include        <string.h>
#include    <stdlib.h>
#include    <time.h>
#include        "TempSub.H"

#ifdef KRL_GNU
  #define get_density_correction_factor get_density_correction_factor__
  #define density_correction density_correction__
  #define nonlinear_density_correction nonlinear_density_correction__
#else
  #define get_density_correction_factor get_density_correction_factor_
  #define density_correction density_correction_
  #define nonlinear_density_correction nonlinear_density_correction_
#endif


extern      StudyPara   ReadTSubDefFile(char*);
extern "C" void get_density_correction_factor(short*, int*, int*, int*,
                                     short*, short*, char*, int*,
                                     short(*)[2], short*);
extern "C" int density_correction(short*, int*, int*, short(*)[2]);

extern "C" void nonlinear_density_correction(short *CurImage,
                                    short *PreImage,
                                    int   *col, int *lin,
                                          char  *DefFile)
{
StudyPara   studyPara = ReadTSubDefFile(DefFile);
if (studyPara.densityCorrection == OFF) {
  printf("DensityCorrection is OFF\n");
  return;
  }

printf("Nonlinear Density Correction\n");
int   grayscale = -1;         // 1:0=Lighter, -1:0=Darker
short dens = 3;         // Scanner Density Range; 3:(0-3)4:(0-4)
short type = 1;         // Screen/Film Type; 1:Med/OC, 2:Med/TMG,
3:insight HC
int   leng = strlen(studyPara.denCorLutDir);
short DCTable[1024][2]; // LUT
short TableNo;

/*---------------------------------------------------------------------
---
      Nonlinear Density Correction for Current Image
-----------------------------------------------------------------------
---*/
get_density_correction_factor(CurImage, col, lin, &grayscale,
                         &dens, &type, studyPara.denCorLutDir, &leng,
                         DCTable, &TableNo);
printf("CurImage; TableNo = %d\n", TableNo);
//if (TableNo != 0 && (TableNo < 8 || TableNo > 9)) {
if (TableNo != 0) {
  (void)density_correction(CurImage, col, lin, DCTable);
  }
else {
  printf("CurImage; Density correction was not necessary.\n");
  }

/*-----------------------------------------------------------------------
---
      Nonlinear Density Correction for Previous Image
-------------------------------------------------------------------------
---*/
get_density_correction_factor(PreImage, col, lin, &grayscale,
                         &dens, &type, studyPara.denCorLutDir, &leng,
                         DCTable, &TableNo);
printf("PreImage; TableNo = %d\n", TableNo);
//if (TableNo != 0 && (TableNo < 8 || TableNo > 9)) {
if (TableNo != 0) {
  (void)density_correction(PreImage, col, lin, DCTable);
  }
else {
  printf("PreImage; Density correction was not necessary.\n");
  }
}
```

```
c       ========================================================
        subroutine nullplus(filename, nchar)
c       To add null character following strings
c       filename        strings                         (IN/OUT)
c       nchar           No. of characters               (OUT)
c
c       JUL 5, 1994     Coded by SHIGE
c       ========================================================
        implicit integer*4 (i-n)
        character*(*)   filename
        character*256   file
        character*1     f(256)
        equivalence     (file, f(1))

        file = filename

        call countchar(file,nchar)
        f(nchar+1) = char(0)

        filename = file

        return
        end
c       =========================================================
        subroutine nullminus(outfile, nchar)
c       Replace Null with Space
c       nchar < 0       NO NULL CHARACTER
c       Coded by SHIGE, 12/27/95
c       ========================================================
        implicit integer*4 (i-n)
        character*(*)   outfile
        character*256   filename
        character*1     file(256)
        equivalence     (filename,file(1))
c
        nb = 256

        filename = outfile

        do i = 1, nb
          if(file(i) .eq. char(0)) goto 11
        end do
        write(*,*)'***** NO NULL CHARATER *****'
        nchar = -1
        return

11      nchar = i - 1

        do j = i, nb
          file(j) = ' '
        end do

        outfile = filename

        return
        end
c       ===============================================================
        subroutine countchar(outfile,nchar)
c       COUNT NO. OF CHARACTERS
c       ===============================================================
        implicit integer*4 (i-n)
        character*(*)   outfile
        character*256   filename
        character*1     file(256)
        equivalence     (filename,file(1))
c
        nb=256
c
        do 10 i=1,nb
                file(i)=' '
10      continue
c
        filename=outfile
c
        do 20 i=nb,1,-1
                if(file(i).ne.' ') goto 21
20      continue
        write(*,*)'***** NO CHARACTERS *****'
        stop
21      nchar=i
        return
        end
```

```
c       ------------------------------------------------------------
c       Quantitative analysis of temporal subtraction image
c       ------------------------------------------------------------
        subroutine quanti2(isub,ncol,nlin,rribcage2,lribcage2,
     &     rribcage_no2,lribcage_no2,midline,top,bot,offset,
     &     iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &     ix1_rule3,imgcontr,imgcontl,iroiIL,shiftmid,ribtop,findi,
     &     magnify)
c       ------------------------------------------------------------
!       Function of this subroutine is to determine the widths of the
!       histograms for right and left lungs.
!       Coded by Takayuki Ishida:  Jan.14.1998

!       (1) First of all, do segmentation of lungs using ribcage edges.
!           Image of ROI "iroiMAP(i,j)" include 4 values.
!           Upper-Right = 1, Lower-Right = 2,
!           Upper-Left  = 3, Lower-Left  = 4
!       (2) Segmentation of lungs using cardiac edges.
!       (3) Determination of final ROI. (size of ROI : 30 x 120 )
!       (4) Calculate average contrast and width of histograms
!           Average contrast is determined by averaging of absolute
!           pixel value of subtraction image.
!           Width of histogram at a 10% of maximum level is determined
!           from smoothed histogram.
c       ------------------------------------------------------------
        implicit   integer*4 (i-n)

        integer*4 ncol,nlin              ! Matix size of the image
[I]
        integer*2 isub(ncol,nlin)        ! input subtraction image
[I]
        integer*2 rribcage2(2,1215)      ! Right Ribcage Points for Image2
[I]
        integer*2 lribcage2(2,1215)      ! Left  Ribcage Points for Image2
[I]
        integer*2 rribcage_no2           ! No. of Right Ribcage Points for Image2
[I]
        integer*2 lribcage_no2           ! No. of Left  Ribcage Points for Image2
[I]
        integer*4 midline,top,bot        ! midline, top-lung, bottom-lung locations
[I]
        integer*4 offset                 ! offset value used for the subtraction
image[I]
!---------------------------------------------------------------------------
        integer*4 ix1_rule2,iy1_rule2    ! Upper-right location of mediastinum
        integer*4 ix2_rule2,iy2_rule2    ! Upper-left of left cardiac edge
        integer*4 ix1_rule3(2,600)       ! x-locations of cardiac edges
                                         ! ix1_rule3(1,600) => right cardiac
x-location
                                         ! ix1_rule3(2,600) => left cardiac
x-location
        integer*4 iy1_rule1              ! bottom of right cardiac edge
!
!       (ix1_rule2,iy1_rule2) --------> x       |
!                                       |       |
!                                       |       |
!                                       |       |
!                                       |       |
!                                       |       x <-------- (ix2_rule2,iy2_rule2)
!                                      /         \
!                                     |           \
!                                     |            \
!                                     |             |
```

```
!               iy1_rule1  --------> x             |
!
!---------------------------------------------------------------------------
        integer*4 imgcontr,imgcontl
        integer*2 iroiIL(ncol,nlin)      ! Illustration of Small ROI
[O]
                                         ! This is an image of segmented ROI
                                         ! for determination of width of the
                                         ! histogram.

        integer*4 shiftmid,ribtop        ! Global shift values for x and y location
[I]
        character*1 findi                ! Indicator for saving file name
[I]
                                         ! (Sub1 or Sub2). Not so important.
        real*4    magnify                ! Contrast factor for subtraction image
[I]

c       ------------------------------------------------------------
!       These two histograms are not used for determination of
!       the width of the histograms. (histogram(-1024:1023) and histoabs(0:1023))
        integer*4 histogram(-1024:1023) ! Histogram of the right and the left lungs
of
                                         ! subtraction image.
        integer*4 histoabs(0:1023)       ! Histogram of the absolute pixel value
                                         ! of the right and the left subtraction
image.
c       ------------------------------------------------------------
        integer*4 val_hist               ! work for making histogram.

        integer*4 histo_R(-1024:1023)    ! Histogram of the right lung for
                                         ! determination of the width of the
histogram.
        integer*4 histo_L(-1024:1023)    ! Histogram of the left lung for
                                         ! determination of the width of the
histogram.

        real*4 histo_R_SM(2048)          ! Smoothed histogram of the right lung
                                         ! determination of the width of the
histogram.
        real*4 histo_L_SM(2048)          ! Smoothed histogram of the left lung
                                         ! determination of the width of the
histogram.
        integer*2 nn,ma                  ! Parameters for smoothing using moving
average.
                                         ! Defined in the smoothing part in this
subroutine.
        integer*4 ix1_rule3wk(2,-100:700) ! work area
!       ------------------------------------------------------------
!       Not used in this subroutine.
        integer*4 iprof(2,600)           ! signature of the small ROIs.
!       ------------------------------------------------------------

        integer*2  iroiMAP(600,600)      ! Image map of Large ROI
c       ------------------------------------------------------------
c       ***************************************************************
!       ------------------------------------------------------------
!       Parameters
!       ------------------------------------------------------------
        itopcut=100              ! un-used margin of top-lungs for small ROI.
        isidecut=30              ! un-used margin belong Rt and Lt ribcage edges
                                 ! for small ROI.
        iheartcut=30             ! un-used margin belong Rt and Lt cardiac edges
```

```
                                 ! for small ROI.
        ibotcut=50               ! un-used margin of bottom-lungs for small ROI.
        ibandwidth=30            ! ROI width
        ibandhight=120           ! ROI hight
        imageavg=0               ! average for the subtraction image.
!       ------------------------------------------------------------
!       Global shift for cardiac edge.
!       ------------------------------------------------------------
        iy1_rule1=iy1_rule1-ribtop
        ix1_rule2=ix1_rule2-shiftmid-iheartcut
        ix2_rule2=ix2_rule2-shiftmid+iheartcut
        iy2_rule2=iy2_rule2-ribtop

        do i=-100,700
          ix1_rule3wk(1,i)=ix1_rule2
          ix1_rule3wk(2,i)=ix2_rule2
        end do

        do i=1,586
          if(ix1_rule3(1,i).ne.0.and.ix1_rule3(2,i).ne.0) then
            ix1_rule3wk(1,i-ribtop)=ix1_rule3(1,i)-shiftmid-iheartcut
            ix1_rule3wk(2,i-ribtop)=ix1_rule3(2,i)-shiftmid+iheartcut
          end if
        end do

!       ------------------------------------------------------------
!       Determine lung bottom location
!       "iy1_rule1" means the lowest location of the right cardiac edge
!       which was detected with a subroutine chs_sub.f.
!       "ibotcut" means un-used margin of bottom-lungs for small ROI.
!       ------------------------------------------------------------
        bot=iy1_rule1-ibotcut

!       ------------------------------------------------------------
!       Determination of the center of the ROI.
!       ------------------------------------------------------------
        ihalf=(bot-top)/2+top+itopcut/2

!       ------------------------------------------------------------
!       Make original-contrast subtraction image for determination of the
!       width of the histogram.
!       And also iniitialization of ROI area map.
!       ------------------------------------------------------------
        do j=1,ncol
          do i=1,nlin
              isub(i,j)=int(real(isub(i,j)-offset)/magnify)
              iroiMAP(i,j)=0
              iroiIL(i,j)=0
          end do
        end do

!       ------------------------------------------------------------
!       Initialization of the histograms.
!       ------------------------------------------------------------
          do i=-1024,1023
              histogram(i)=0
              histo_R(i)=0
              histo_L(i)=0
          end do
          do i=0,1023
              histoabs(i)=0
          end do
          do i=1,2048
```

```
              histo_R_SM(i)=0.0
              histo_L_SM(i)=0.0
          end do

c       ------------------------------------------------------------
c       Lung segmentation for Large ROIs using ribcage edges.
c       Upper-Right     : iroiMAP(i,j)=1
c       Lower-Right     : iroiMAP(i,j)=2
c       Upper-Left      : iroiMAP(i,j)=3
c       Lower-Left      : iroiMAP(i,j)=4
c       ------------------------------------------------------------
          do k=itopcut,rribcage_no2
             do l=rribcage2(1,k)+isidecut,midline
               if(rribcage2(2,k).lt.ihalf) then
                  iroiMAP(l,rribcage2(2,k))=1
               end if
        if(rribcage2(2,k).ge.ihalf.and.rribcage2(2,k).lt.bot) then
                  iroiMAP(l,rribcage2(2,k))=2
               end if
             end do
          end do

          do k=itopcut,lribcage_no2
             do l=midline+1,lribcage2(1,k)-isidecut
               if(lribcage2(2,k).lt.ihalf) then
                 iroiMAP(l,lribcage2(2,k))=3
               end if
         if(lribcage2(2,k).ge.ihalf.and.lribcage2(2,k).lt.bot) then
                 iroiMAP(l,lribcage2(2,k))=4
               end if
             end do
          end do

          do k=itopcut,rribcage_no2
             ict=rribcage2(2,k)-rribcage2(2,k-1)
             if(ict.gt.1) then
             do ii=1,ict
             do l=rribcage2(1,k-1)+isidecut,midline
               if(rribcage2(2,k).lt.ihalf) then
                  iroiMAP(l,rribcage2(2,k)-ii)=1
               end if
        if(rribcage2(2,k).ge.ihalf.and.rribcage2(2,k).lt.bot) then
                  iroiMAP(l,rribcage2(2,k)-ii)=2
               end if
             end do
             end do
             end if
          end do
          do k=itopcut,lribcage_no2
             ict=lribcage2(2,k)-lribcage2(2,k-1)
             if(ict.gt.1) then
             do ii=1,ict
             do l=midline+1,lribcage2(1,k)-isidecut
               if(lribcage2(2,k).lt.ihalf) then
                  iroiMAP(l,lribcage2(2,k)-ii)=3
               end if
         if(lribcage2(2,k).ge.ihalf.and.lribcage2(2,k).lt.bot) then
                 iroiMAP(l,lribcage2(2,k)-ii)=4
               end if
             end do
             end do
               end if
          end do
```

```
c      -------------------------------------------------------------
c      Ignore mediastinum area and below diaphragm.
c      -------------------------------------------------------------
       do j=1,ncol
         do i=1,nlin

               if(i.ge.ix1_rule2.and.i.lt.ix2_rule2.
     &             and.j.lt.iy2_rule2) then
                    iroiMAP(i,j)=0
               end if


               if(j.ge.iy2_rule2.and.j.lt.iy1_rule1.
     &             and.i.ge.ix1_rule3wk(1,j).and.
     &             i.lt.ix1_rule3wk(2,j)) then
                       iroiMAP(i,j)=0
               end if
         end do
       end do

c        call writeimage(iroiMAP,'temp.img' ,586,586)

c      -------------------------------------------------------------
c      Determination of band-shape small ROIs for determination of
c      the width of the histogram.
c      -------------------------------------------------------------
c      --------------------------------------------------------------------
c      This part is for determination of top and bottom limit of the ROI.
c      --------------------------------------------------------------------
       if(iy1_rule1-rribcage2(2,itopcut).lt.ibandhight) then
           iystart=rribcage2(2,itopcut)
           iyend=iy1_rule1
       else
           iystart=ihalf-ibandhight/2
           iyend=ihalf+ibandhight/2
       end if
         do j=1,iystart-1
           do i=1,midline
               iroiMAP(i,j)=0
           end do
         end do
         do j=iyend+1,ncol
           do i=1,midline
               iroiMAP(i,j)=0
           end do
         end do

c      -------------------------------------------------------------
c      This part is for determination of Rt and Lt limit of the ROI.
c      -------------------------------------------------------------
       do j=iystart,iyend
               do k=1,ix1_rule3wk(1,j)
                if(iroiMAP(k,j).eq.1.or.iroiMAP(k,j).eq.2) then
                     ist=k
                     goto 446
                end if
               end do
 446            ix_cen=(ix1_rule3wk(1,j)-ist)/2+ist
               do k=ist,ix_cen-ibandwidth/2
                      iroiMAP(k,j)=0
               end do
               do k=ix_cen+ibandwidth/2,ix1_rule3wk(1,j)
```

```
                      iroiMAP(k,j)=0
               end do
       end do

c      -------------------------------------------------------------
c      Hight of the ROI in the "iroiMAP" =< 120
c      -------------------------------------------------------------
       if(iy1_rule1-lribcage2(2,itopcut).lt.ibandhight) then
           iystart=lribcage2(2,itopcut)
           iyend=iy1_rule1
       else
           iystart=ihalf-ibandhight/2
           iyend=ihalf+ibandhight/2
       end if
         do j=1,iystart-1
           do i=midline+1,ncol
               iroiMAP(i,j)=0
           end do
         end do
         do j=iyend+1,ncol
           do i=midline+1,ncol
               iroiMAP(i,j)=0
           end do
         end do
       do j=iystart,iyend
               do k=ix1_rule3wk(2,j)+1,ncol
                   if(iroiMAP(k,j).eq.0) then
                     ist=k-1
                     goto 447
                   end if
               end do
 447           ix_cen=(ist-ix1_rule3wk(2,j))/2+ix1_rule3wk(2,j)
               do k=ix1_rule3wk(2,j),ix_cen-ibandwidth/2
                      iroiMAP(k,j)=0
               end do
               do k=ix_cen+ibandwidth/2,ist
                      iroiMAP(k,j)=0
               end do
       end do
c         call quanti2_prof(isub,ncol,nlin,iroiMAP,iprof,findi)
c        call writeimage(iroiMAP,'temp.img' ,586,586)
c      -------------------------------------------------------------
c      Determine average contrast and SD of the subtraction image.
c      Determine Histograms width of the subtraction image.
c      The average contrast and SD of the subtraction image are determined
c      here, however, the average contrast and SD are not used.
c
c      Histograms of subtraction image are determined here.
c      -------------------------------------------------------------
       abs_pixa=0.0
       abs_pixi=0.0
       abs_pixo=0.0
       abs_pixr=0.0
       abs_pixl=0.0
       abs_pixq1=0.0
       abs_pixq2=0.0
       abs_pixq3=0.0
       abs_pixq4=0.0
       abs_sda=0.0
       abs_sdi=0.0
       abs_sdo=0.0
       abs_sdr=0.0
       abs_sdl=0.0
```

```
       abs_sdq1=0.0
       abs_sdq2=0.0
       abs_sdq3=0.0
       abs_sdq4=0.0
       numa=0
       numo=0
       numi=0
       numr=0
       numl=0
       numq1=0
       numq2=0
       numq3=0
       numq4=0
          do j=1,nlin
            do i=1,ncol
               if(iroiMAP(i,j).ge.0) then
               numa=numa+1
               abs_pixa=abs_pixa+abs(isub(i,j)-imageavg)
               abs_sda=abs_sda+abs(isub(i,j)-imageavg)**2
               end if
               if(iroiMAP(i,j).eq.0) then
               numo=numo+1
               abs_pixo=abs_pixo+abs(isub(i,j)-imageavg)
               abs_sdo=abs_sdo+abs(isub(i,j)-imageavg)**2
               end if
               if(iroiMAP(i,j).ge.1) then
               numi=numi+1
               val_hist=isub(i,j)
               histogram(val_hist)=histogram(val_hist)+1
               abs_pixi=abs_pixi+abs(isub(i,j)-imageavg)
               abs_sdi=abs_sdi+abs(isub(i,j)-imageavg)**2
               end if
c      -------------------------------------------------------------
c      This part is important for this subroutine.
!      Detemination of histograms of right and left lung.
c      -------------------------------------------------------------
               if(iroiMAP(i,j).eq.1.or.
     &            iroiMAP(i,j).eq.2) then
               numr=numr+1
               val_hist=isub(i,j)
               histo_R(val_hist)=histo_R(val_hist)+1
               abs_pixr=abs_pixr+abs(isub(i,j)-imageavg)
               abs_sdr=abs_sdr+abs(isub(i,j)-imageavg)**2
               end if
               if(iroiMAP(i,j).eq.3.or.
     &            iroiMAP(i,j).eq.4) then
               numl=numl+1
               val_hist=isub(i,j)
               histo_L(val_hist)=histo_L(val_hist)+1
               abs_pixl=abs_pixl+abs(isub(i,j)-imageavg)
               abs_sdl=abs_sdl+abs(isub(i,j)-imageavg)**2
               end if
c      -------------------------------------------------------------
               if(iroiMAP(i,j).eq.1) then
               numq1=numq1+1
               abs_pixq1=abs_pixq1+abs(isub(i,j)-imageavg)
               abs_sdq1=abs_sdq1+abs(isub(i,j)-imageavg)**2
               end if
               if(iroiMAP(i,j).eq.2) then
               numq2=numq2+1
               abs_pixq2=abs_pixq2+abs(isub(i,j)-imageavg)
               abs_sdq2=abs_sdq2+abs(isub(i,j)-imageavg)**2
               end if
```

```
               if(iroiMAP(i,j).eq.3) then
               numq3=numq3+1
               abs_pixq3=abs_pixq3+abs(isub(i,j)-imageavg)
               abs_sdq3=abs_sdq3+abs(isub(i,j)-imageavg)**2
               end if
               if(iroiMAP(i,j).eq.4) then
               numq4=numq4+1
               abs_pixq4=abs_pixq4+abs(isub(i,j)-imageavg)
               abs_sdq4=abs_sdq4+abs(isub(i,j)-imageavg)**2
               end if
            end do
          end do

            abs_pixa=abs_pixa/real(numa)       !Avg. contrast of all image
            abs_pixo=abs_pixo/real(numo)       !Avg. contrast of outside ribcage
            abs_pixi=abs_pixi/real(numi)       !Avg. contrast of inside ribcage

            abs_pixr=abs_pixr/real(numr)       !Avg. contrast of right lung
            abs_pixl=abs_pixl/real(numl)       !Avg. contrast of left lung

            abs_pixq1=abs_pixq1/real(numq1)    !Avg. contrast of UR lung
            abs_pixq2=abs_pixq2/real(numq2)    !Avg. contrast of LR lung
            abs_pixq3=abs_pixq3/real(numq3)    !Avg. contrast of UL lung
            abs_pixq4=abs_pixq4/real(numq4)    !Avg. contrast of LL lung

            abs_sda=sqrt(abs_sda/real(numa))    !SD of histogram for all image
            abs_sdo=sqrt(abs_sdo/real(numo))    !SD of histogram for outside
ribcage
            abs_sdi=sqrt(abs_sdi/real(numi))    !SD of histogram for inside ribcage

            abs_sdr=sqrt(abs_sdr/real(numr))    !SD of histogram for right lung
            abs_sdl=sqrt(abs_sdl/real(numl))    !SD of histogram for left lung

            abs_sdq1=sqrt(abs_sdq1/real(numq1))!SD of histogram for UR lung
            abs_sdq2=sqrt(abs_sdq2/real(numq2))!SD of histogram for LR lung
            abs_sdq3=sqrt(abs_sdq3/real(numq3))!SD of histogram for UL lung
            abs_sdq4=sqrt(abs_sdq4/real(numq4))!SD of histogram for LL lung

c          write(*,5100) abs_pixr,abs_pixl,max(abs_pixr,abs_pixl)
c 5100     format ('respix',3f9.3)
c          write(*,5101) abs_sdr,abs_sdl
c 5101     format ('ressd',2f9.3)

c      ----------------------------------------------------------
c       SMOOTHING HISTOGRAMS BY USING RUNING AVERAGE METHOD
c      ----------------------------------------------------------
       ict=0
       do i=-1024,1023
           ict=ict+1
              histo_R_SM(ict)=histo_R(i)
              histo_L_SM(ict)=histo_L(i)
       end do
       nn=ict
       ma=11
       do i=1,2
          call smooth(histo_R_SM,nn,ma)
          call smooth(histo_L_SM,nn,ma)
       end do
       ict=0
       do i=6,2044
           ict=i-1024
              histo_R(ict)=int(histo_R_SM(i))
              histo_L(ict)=int(histo_L_SM(i))
```

```
        end do
c        ----------------------------------------------------------
c           Determine the width of the histogram for right lung
c        ----------------------------------------------------------
c       ********************************************
c        Get maximum of a histogram for right lung
c       ********************************************
                 max_hist=-10
                 max_x=0
c          open (1,file=findi//'_hissub')
c               write(1,*) '"right'
           do i=-1020,1020
c               write(1,*) i,histo_R(i)
               if(histo_R(i).gt.max_hist) then
                    max_hist=histo_R(i)
                    max_x=i
               end if
           end do
c       **********************************
c         Determine width at 10% peak
c       **********************************
           ihw1=0
           ihw2=0
           wid=0.10
           do i=max_x,1020
               if(histo_R(i).lt.int(real(max_hist)*wid)) then
                    ihw1=i
                    goto 12
               end if
           end do
 12         do i=max_x,-1020,-1
               if(histo_R(i).lt.int(real(max_hist)*wid)) then
                    ihw2=i
                    goto 13
               end if
           end do
 13         ifwhmr=ihw1-ihw2
c          write(*,*) 'reswidth_R',max_x,max_hist,ihw2,ihw1,ifwhmr
c        ----------------------------------------------------------
c          Determine the width of the histogram for left lung
c        ----------------------------------------------------------
c       ********************************************
c        Get maximum of a histogram for left lung
c       ********************************************
                 max_hist=-10
                 max_x=0
c                write(1,*) ' '
c                write(1,*) '"left '
           do i=-1020,1020
c               write(1,*) i,histo_L(i)
               if(histo_L(i).gt.max_hist) then
                    max_hist=histo_L(i)
                    max_x=i
               end if
           end do
c         close(1)
c       *********************************************
c         Determination of width of the histograms
c       *********************************************
           ihw1=1014
           ihw2=-1024
           do i=max_x,1020
               if(histo_L(i).lt.int(real(max_hist)*wid)) then
```

```
                    ihw1=i
                    goto 14
               end if
           end do
 14         do i=max_x,-1020,-1
               if(histo_L(i).lt.int(real(max_hist)*wid)) then
                    ihw2=i
                    goto 15
               end if
           end do
 15         ifwhml=ihw1-ihw2
c          write(*,*) 'reswidth_L',max_x,max_hist,ihw2,ihw1,ifwhml
c          write(*,*) 'reswidth_small',ifwhmr,ifwhml,max(ifwhmr,ifwhml)
c        ----------------------------------------------------------
c           Determination of final results of width of the histogram
c           Width of the histogram for right lung = imgcontr
c           Width of the histogram for left lung  = imgcontl
c        ----------------------------------------------------------
          imgcontr=ifwhmr
          imgcontl=ifwhml

c        ----------------------------------------------------------
!        Remake magnified-contrast subtraction image.
c        ----------------------------------------------------------
          do j=1,ncol
            do i=1,nlin
               isub(i,j)=int(real(isub(i,j)*magnify))+offset
            end do
          end do

c        ----------------------------------------------------------
!        Make a skelton image of ROIs
c        ----------------------------------------------------------
          do j=1,ncol
            do i=1,nlin
               if(iroiMAP(i,j).gt.0) iroiMAP(i,j)=1
            end do
          end do
          call kyoukai(iroiMAP,iroiIL,ncol,nlin,1)

c        ----------------------------------------------------------
!        Reset location for the cardiac land-marks
c        ----------------------------------------------------------
          iy1_rule1=iy1_rule1+ribtop
          ix1_rule2=ix1_rule2+shiftmid+iheartcut
          ix2_rule2=ix2_rule2+shiftmid-iheartcut
          iy2_rule2=iy2_rule2+ribtop

        return
        end
```

```
/*=================================================================
      Subroutine read_images_skip
      Read FCR Image and Reduce by RATIO
      CurImage    Current  Image reduced by RATIO      [O]
      PreImage    Previous Image reduced by RATIO      [O]
      p_col, p_lin      Matrix Size of CurImage and PreImage[O]
      CurFName    Current  Image Name             [I]
      PreFName    Previous Image Name             [I]
      Coded by SHIGE 4/16/97
      Modified by SHIGE 10/28/98 for DICOM
      Modified by Roger 05/24/99 for PC-LINUX
==================================================================*/
#include    <stdio.h>
#include        <string.h>
#include    <stdlib.h>
#include        "TempSub.H"

#ifdef KRL_GNU                // KRL_GNU is defined by Makefile if PC-
LINUX
  #define read_images_skip read_images_skip__
#else
  #define read_images_skip read_images_skip_
#endif

InputImageFormat  getImageFormat(char*, int*, int*);
void        GetFImage(short*, char,  int, int, int, short*, int*, int*);
void        GetDImage(short*, char*, int, int, int, short*, int*, int*);
void        GetTImage(short*, ThvDicom, int, int, float, short*, int*,
int*);
void        GetNImage(short*, int, int, int, short*, int*, int*);
extern            StudyPara ReadTSubDefFile(char*);
extern            FcrStand  getFcrStandInfo(FILE*, char*);
extern            FcrDicom  getFcrDicomInfo(FILE*, char*);
extern            ThvDicom  getThvDicomInfo(FILE*, char*);

extern "C" int read_images_skip(short *CurImage,       short *PreImage,
                                int   *p_col,        int   *p_lin,
                          char  *DefFile,
                                char  *PreFName,     char  *CurFName)
{
FILE        *fp;
StudyPara   studyPara = ReadTSubDefFile(DefFile);
short       *CurOrgImage, *PreOrgImage;
int         mszx, mszy, ratio;            // for Original Image
int         PreMszx, PreMszy;       // for Thoravision
int         CurMszx, CurMszy;
float       fratio;                       // for Thoravision
char        CurExamDate[60], PreExamDate[60];
char        *FcrStdHeader;
char        *FcrDcmHeader;
char        *ThvDcmHeader;

FcrStand        CurFcr, PreFcr;
FcrDicom        CurDcm, PreDcm;
ThvDicom    CurThv, PreThv;

int         msz;                    // for Thoravision
int         mx, my, i;              // for Thoravision

/*----------------------------------------------------------------
      Check Image Format
```

```
----------------------------------------------------------------*/
if ((studyPara.inputImageFormat != getImageFormat(CurFName, &mszx,
&mszy) ||
    studyPara.inputImageFormat != getImageFormat(PreFName, &mszx,
&mszy)) &&
    (studyPara.inputImageFormat != INPUT_NON_HEADER))
   {
   printf("\a\aWrong inputImageFormat in %s\n", DefFile); exit(0);
   }

if (studyPara.inputImageFormat == INPUT_STANDARD_FCR_HEADER) {
  printf("\a\aINPUT_STANDARD_FCR_HEADER is not permitted\n");
  exit(0);
  }

else if (studyPara.inputImageFormat == INPUT_DICOM_FCR_HEADER) {
  /*--------------------------------------------------------------
     Memory Allocation for Original FCR Images [DICOM FCR]
  --------------------------------------------------------------*/
  FcrDcmHeader = new char[MAX_FCR_DCM_HEADER_SIZE];
  if (mszy > FCR_WY) mszy = FCR_WY;
  if (mszx == FCR_WX / RATIO) ratio = 1;
  else                        ratio = RATIO;
  CurOrgImage = new short[mszx * mszy];
  PreOrgImage = new short[mszx * mszy];

  /*--------------------------------------------------------------
      Read Original FCR Images (Current) [DICOM FCR]
  ---------------------------------------------------------------*/
  if ((fp = fopen(CurFName,"r")) == NULL) {
    printf("%s is missing\n", CurFName); exit(0);
    }
  CurDcm = getFcrDicomInfo(fp, FcrDcmHeader);
  if (0 != strcmp(CurDcm.IPSize, DAIKAKU_DCM) &&
      0 != strcmp(CurDcm.IPSize, HANSETU_DCM) &&
      0 != strcmp(CurDcm.IPSize, ONE_THIRD_DCM)) {
    printf("Wrong IP Size in Current Image\n"); exit(0);
    }
  else if (0 == strcmp(CurDcm.IPSize, HANSETU_DCM)) {
    printf("IP Size of Current Image is 14 x 17\n");
    if (0 == strcmp(CurDcm.Dir,  V_FLIP_DCM) ||
      0 == strcmp(CurDcm.Dir, HV_FLIP_DCM))
        fseek(fp, sizeof(short) * mszx * SKIP_HANSETU, SEEK_CUR);
    }
  if (0 != strcmp(CurDcm.Dir, H_FLIP_DCM) && 0 != strcmp(CurDcm.Dir,
NO_FLIP_DCM) &&
      0 != strcmp(CurDcm.Dir, V_FLIP_DCM) && 0 != strcmp(CurDcm.Dir,
HV_FLIP_DCM)) {
      printf("Wrong IP Scanning Direction\n"); exit(0);
      }
  #ifdef     _SUN_SPARC_SOLARIS_1X_
    fread((char*)CurOrgImage, sizeof(short), mszx * mszy, fp);
  #else
    fread(CurOrgImage, sizeof(short), mszx * mszy, fp);
  #endif
  fclose(fp);
  sprintf(CurExamDate, "%s%s", CurDcm.Date, CurDcm.Time);

  /*--------------------------------------------------------------
      Read Original FCR Images (Previous) [DICOM FCR]
  ---------------------------------------------------------------*/
```

```
  if ((fp = fopen(PreFName,"r")) == NULL) {
    printf("%s is missing\n", PreFName); exit(0);
    }
  PreDcm = getFcrDicomInfo(fp, FcrDcmHeader);
  if (0 != strcmp(CurDcm.IPSize, DAIKAKU_DCM) &&
      0 != strcmp(CurDcm.IPSize, HANSETU_DCM) &&
      0 != strcmp(CurDcm.IPSize, ONE_THIRD_DCM)) {
    printf("Wrong IP Size in Current Image\n"); exit(0);
    }
  else if (0 == strcmp(PreDcm.IPSize, HANSETU_DCM)) {
    printf("IP Size of Previous Image is 14 x 17\n");
    if (0 == strcmp(PreDcm.Dir,  V_FLIP_DCM) ||
      0 == strcmp(PreDcm.Dir, HV_FLIP_DCM))
        fseek(fp, sizeof(short) * mszx * SKIP_HANSETU, SEEK_CUR);
    }
  if (0 != strcmp(PreDcm.Dir, H_FLIP_DCM) && 0 != strcmp(PreDcm.Dir,
NO_FLIP_DCM) &&
      0 != strcmp(PreDcm.Dir, V_FLIP_DCM) && 0 != strcmp(PreDcm.Dir,
HV_FLIP_DCM)) {
      printf("Wrong IP Scanning Direction\n"); exit(0);
      }
  #ifdef     _SUN_SPARC_SOLARIS_1X_
    fread((char*)PreOrgImage, sizeof(short), mszx * mszy, fp);
  #else
    fread(PreOrgImage, sizeof(short), mszx * mszy, fp);
  #endif
  fclose(fp);
  sprintf(PreExamDate, "%s%s", PreDcm.Date, PreDcm.Time);

  /*--------------------------------------------------------------
      Check Pair of Same Patient [DICOM FCR]
  ---------------------------------------------------------------*/
  if (!(strcmp(CurExamDate,  PreExamDate )  > 0 &&
      strcmp(CurDcm.PatID, PreDcm.PatID) == 0)) {
    printf("Wrong Image Pair!!!\n");
    printf("\tPatientID1 : %s\tPatientID2 : %s\n", CurDcm.PatID,
PreDcm.PatID);
    printf("\tCurExamDate  : %s\tPreExamDate  : %s\n", CurExamDate,
PreExamDate);
    exit(0);
    }

  /*--------------------------------------------------------------
      Reduction, Flip and Inversion for FCR Image [DICOM FCR]
  ---------------------------------------------------------------*/
  *p_col = mszx / ratio;
  *p_lin = mszy / ratio;
  GetDImage(CurOrgImage, CurDcm.Dir, mszx, mszy, ratio, CurImage, p_col,
p_lin);
  GetDImage(PreOrgImage, PreDcm.Dir, mszx, mszy, ratio, PreImage, p_col,
p_lin);
  printf("DICOM FCR : (%d, %d)===>(%d, %d)\n", mszx, mszy, *p_col,
*p_lin);
  delete FcrDcmHeader;
  }
else if (studyPara.inputImageFormat == INPUT_DICOM_THV_HEADER) {
  /*------------------------------------------------------------
      Read Current Image [Thoravision]
  -------------------------------------------------------------*/
  ThvDcmHeader = new char[MAX_THV_HEADER_SIZE];
  if ((fp = fopen(CurFName,"r")) == NULL) {
```

```
    printf("%s is missing\n", CurFName); exit(0);
    }
  CurThv = getThvDicomInfo(fp, ThvDcmHeader);
  sprintf(CurExamDate, "%s%s", CurThv.Date, CurThv.Time);
  CurMszx = CurThv.Col;
  CurMszy = CurThv.Row;
  CurOrgImage = new short[CurMszx * CurMszy];
  #ifdef _SUN_SPARC_SOLARIS_1X_
    fread((char*)CurOrgImage, sizeof(short), CurMszx * CurMszy, fp);
  #else
    fread(CurOrgImage, sizeof(short), CurMszx * CurMszy, fp);
  #endif
  fclose(fp);

  /*--------------------------------------------------------------
      Read Previous Image [Thoravision]
  ---------------------------------------------------------------*/
  if ((fp = fopen(PreFName,"r")) == NULL) {
    printf("%s is missing\n", PreFName); exit(0);
    }
  PreThv = getThvDicomInfo(fp, ThvDcmHeader);
  sprintf(PreExamDate, "%s%s", PreThv.Date, PreThv.Time);
  PreMszx = PreThv.Col;
  PreMszy = PreThv.Row;
  PreOrgImage = new short[PreMszx * PreMszy];
  #ifdef     _SUN_SPARC_SOLARIS_1X_
    fread((char*)PreOrgImage, sizeof(short), PreMszx * PreMszy, fp);
  #else
    fread(PreOrgImage, sizeof(short), PreMszx * PreMszy, fp);
  #endif
  fclose(fp);

  /*--------------------------------------------------------------
      Check Exam Date [Thoravision]
  ---------------------------------------------------------------*/
  if (!(strcmp(CurExamDate,  PreExamDate )  > 0)) {
    printf("\a\aWrong Image Pair!!!\n");
    printf("CurExamDate must be larger than PreExamDate\n");
    printf("\tCurExamDate  : %s\tPreExamDate  : %s\n", CurExamDate,
PreExamDate);
    exit(0);
    }

  /*--------------------------------------------------------------
      Determination of Matrix Size [Thoravision]
  ---------------------------------------------------------------*/
  (CurMszx >= PreMszx) ? (mx = CurMszx) : (mx = PreMszx);
  (CurMszy >= PreMszy) ? (my = CurMszy) : (my = PreMszy);
  (mx >= my) ? (msz = mx) : (msz = my);

  for (i = 0; i < 10; i++) {
    fratio = FRATIO + (float)i * 0.1;
    *p_col = *p_lin = (float)msz / fratio;
    if (*p_col <= 600) break;
    }

  printf("THV      : Reduction Ratio = %f\n", fratio);
  printf("Current : (%d, %d)===>(%d, %d)\n", CurMszx, CurMszy, *p_col,
*p_lin);
  printf("Previous: (%d, %d)===>(%d, %d)\n", PreMszx, PreMszy, *p_col,
*p_lin);
```

```
  /*------------------------------------------------------------------
      Reduction, Flip, Inversion and Padding[Thoravision]
  ------------------------------------------------------------------*/
  GetTImage(CurOrgImage, CurThv, CurMszx, CurMszy, fratio, CurImage,
p_col, p_lin);
  GetTImage(PreOrgImage, PreThv, PreMszx, PreMszy, fratio, PreImage,
p_col, p_lin);
  delete ThvDcmHeader;
  }
else if (studyPara.inputImageFormat == INPUT_NON_HEADER) {
  /*------------------------------------------------------------------
      Determination of Reduction ratio [Non-Header]
  ------------------------------------------------------------------*/
  mszx = studyPara.col;
  mszy = studyPara.lin;
  (mszx <= mszy) ? (msz = mszx) : (msz = mszy);
  if      (msz >=  500 && msz <=  600) ratio = 1;
  else if (msz >= 1000 && msz <= 1200) ratio = 2;
  else if (msz >= 1500 && msz <= 1800) ratio = 3;
  else if (msz >= 2000 && msz <= 2400) ratio = 4;
  else if (msz >= 4000 && msz <= 4800) ratio = 8;
  else {
    printf("Wrong Matrix Size in %s.\n", DefFile); exit(0);
    }
#if 0
  *p_col = *p_lin = msz / ratio;
#else
  *p_col = mszx / ratio;
  *p_lin = mszy / ratio;
#endif
  CurOrgImage = new short[mszx * mszy];
  PreOrgImage = new short[mszx * mszy];

  /*----------------------------------------------------------------
      Read Current Image [Non-Header]
  ----------------------------------------------------------------*/
  if ((fp = fopen(CurFName,"r")) == NULL) {
    printf("%s is missing\n", CurFName); exit(0);
    }
  #ifdef _SUN_SPARC_SOLARIS_1X_
    fread((char*)CurOrgImage, sizeof(short), mszx * mszy, fp);
  #else
    fread(CurOrgImage, sizeof(short), mszx * mszy, fp);
  #endif
  fclose(fp);

  /*----------------------------------------------------------------
      Read Previous Image [Non-Header]
  ----------------------------------------------------------------*/
  if ((fp = fopen(PreFName,"r")) == NULL) {
    printf("%s is missing\n", PreFName); exit(0);
    }
  #ifdef _SUN_SPARC_SOLARIS_1X_
    fread((char*)PreOrgImage, sizeof(short), mszx * mszy, fp);
  #else
    fread(PreOrgImage, sizeof(short), mszx * mszy, fp);
  #endif
  fclose(fp);

  /*------------------------------------------------------------------
```

```
      Reduction for Original Image [Non-Header]
  ------------------------------------------------------------------*/
  GetNImage(CurOrgImage, mszx, mszy, ratio, CurImage, p_col, p_lin);
  GetNImage(PreOrgImage, mszx, mszy, ratio, PreImage, p_col, p_lin);
  }
else {
  printf("Wrong InputImageFormat in %s.\n", DefFile); exit(0);
  }

/*----------------------------------------
      RETURN
----------------------------------------*/
delete CurOrgImage;
delete PreOrgImage;
return(1);
}

/*=================================================================
      Subroutine GetFImage
      Reduction, Flip and Inversion for FCR Image
=================================================================*/
void  GetFImage(short *OrgImage, char IP_disp,
              int mszx, int mszy, int ratio,
              short *image, int *p_col, int *p_lin)
{
}

/*=================================================================
      Subroutine GetDImage
      Reduction, Flip and Inversion for FCR Image
=================================================================*/
void  GetDImage(short *OrgImage, char *IP_disp,
              int mszx, int mszy, int ratio,
              short *image, int *p_col, int *p_lin)
{
  /* With this file format, pixels should be stored as low-endian.
Thus,
     if the machine is high endian, we need to byte swap. */
  int shouldByteSwap = !_isMachineLowEndian;

  int i, j, k, l, x0, y0, x1, y1;
  short     MaxGray;

  /*------------------------------------------------------
      Reduction and Horizontal/Vertical Flip
  ------------------------------------------------------*/
  MaxGray = 1023;
  if (0 == strcmp(IP_disp, H_FLIP_DCM)) {
    for (j = 0; j < *p_lin; j++) {
      y0 = j * ratio;
      y1 = j;
    for (i = 0; i < *p_col; i++) {
      x0 = i * ratio;
      x1 = *p_col - i - 1;
      k  = y0 * mszx   + x0;
      l  = y1 * *p_col + x1;
      if (shouldByteSwap)
      image[l] = ((OrgImage[k] >> 8 & 0x00FF) | (OrgImage[k] << 8)) ^
MaxGray;
      else
      image[l] = OrgImage[k] ^ MaxGray;
```

```
      }
      }
    }
  else if (0 == strcmp(IP_disp, NO_FLIP_DCM)) {
    for (j = 0; j < *p_lin; j++) {
      y0 = j * ratio;
    for (i = 0; i < *p_col; i++) {
      x0 = i * ratio;
      k  = y0 * mszx   + x0;
      l  = j  * *p_col + i;
      if (shouldByteSwap)
      image[l] = ((OrgImage[k] >> 8 & 0x00FF) | (OrgImage[k] << 8)) ^
MaxGray;
      else
      image[l] = OrgImage[k] ^ MaxGray;
      }
      }
    }
  else if (0 == strcmp(IP_disp, V_FLIP_DCM)) {
    for (j = 0; j < *p_lin; j++) {
      y0 = j * ratio;
      y1 = *p_lin - j - 1;
    for (i = 0; i < *p_col; i++) {
      x0 = i * ratio;
      x1 = i;
      k  = y0 * mszx   + x0;
      l  = y1 * *p_col + x1;
      if (shouldByteSwap)
      image[l] = ((OrgImage[k] >> 8 & 0x00FF) | (OrgImage[k] << 8)) ^
MaxGray;
      else
      image[l] = OrgImage[k] ^ MaxGray;
      }
      }
    }
  else if (0 == strcmp(IP_disp, HV_FLIP_DCM)) {
    for (j = 0; j < *p_lin; j++) {
      y0 = j * ratio;
      y1 = *p_lin - j - 1;
    for (i = 0; i < *p_col; i++) {
      x0 = i * ratio;
      x1 = *p_col - i - 1;
      k  = y0 * mszx   + x0;
      l  = y1 * *p_col + x1;
      if (shouldByteSwap)
      image[l] = ((OrgImage[k] >> 8 & 0x00FF) | (OrgImage[k] << 8)) ^
MaxGray;
      else
      image[l] = OrgImage[k] ^ MaxGray;
      }
      }
    }
  else {
    printf("Wrong IP Scanning Direction\n"); exit(0);
    }
}

/*=================================================================
      Subroutine GetTImage
      Reduction, Flip, Inversion and Padding for Thoravision
=================================================================*/
```

```
void  GetTImage(short *OrgImage, ThvDicom thv,
              int mszx, int mszy, float ratio,
              short *image, int *p_col, int *p_lin)
{
  /* With this file format, pixels should be stored as low-endian.
Thus,
     if the machine is high endian, we need to byte swap. */
  int shouldByteSwap = !_isMachineLowEndian;

short MaxGray = 1023;
short val;
int   Shift   = thv.NBit - 10;
short *buf;
int   mx, my;
int     i, j, ix, iy, jx, jy, k, l, u, v;
int   sx, sy;

/*----------------------------------------
      BEGIN
----------------------------------------*/
mx  = (float)mszx / ratio;
my  = (float)mszy / ratio;
buf = new short[mx * my];

/*----------------------------------------
      Reduce Image
----------------------------------------*/
if (0 == strcmp(thv.Dir, AP_THV)) {
  for (j = 0; j < my; j++) {
    iy = (float)j * ratio + 0.5;
    jy = j;
  for (i = 0; i < mx; i++) {
    ix = (float)i * ratio + 0.5;
    jx = mx - i - 1;
    u  = jy * mx   + jx;
    v  = iy * thv.Col + ix;
    if (shouldByteSwap)
      val    = (OrgImage[v] >> 8 & 0x00FF) | (OrgImage[v] << 8 &
0xFF00);
    else
      val    = OrgImage[v];
    buf[u] = (val >> Shift) ^ MaxGray;
    }
    }
  }
else {
  for (j = 0; j < my; j++) {
    iy = (float)j * ratio + 0.5;
  for (i = 0; i < mx; i++) {
    ix = (float)i * ratio + 0.5;
    u  = j * mx    + i;
    v  = iy * thv.Col + ix;
    if (shouldByteSwap)
      val    = (OrgImage[v] >> 8 & 0x00FF) | (OrgImage[v] << 8 &
0xFF00);
    else
      val    = OrgImage[v];
    buf[u] = (val >> Shift) ^ MaxGray;
    }
    }
  }
/*-------------------------------------------------------
      Copy
--------------------------------------------------------*/
sx = (*p_col - mx) / 2;
for (j = 0; j < my; j++) {
for (i = 0; i < mx; i++) {
  u  = j * *p_col + sx + i;
  v  = j * mx      + i;
  image[u] = buf[v];
  }
  }

/*-------------------------------------------------------
      Padding
--------------------------------------------------------*/
for (j = 0; j < my; j++) {
  for (i = 0; i < sx; i++) {
    v = j * *p_col + sx;
    u = j * *p_col + i;
    image[u] = image[v];
    }
  for (i = sx + mx; i < *p_col; i++) {
    v = j * *p_col + sx + mx - 1;
    u = j * *p_col + i;
    image[u] = image[v];
    }
  }

for (i = 0; i < *p_col; i++) {
  for (j = my; j < *p_lin; j++) {
    v = (my - 1) * *p_col + i;
    u =  j * *p_col + i;
    image[u] = image[v];
    }
  }

delete buf;
}

/*==========================================================
      Subroutine GetNImage
      Reduction for Non-Header Image
===========================================================*/
void  GetNImage(short *OrgImage, int mszx, int mszy, int ratio,
               short *image, int *p_col, int *p_lin)
{
  /* With this file format, pixels should be stored as high-endian.
Thus,
     if the machine is low endian, we need to byte swap. */
  int shouldByteSwap = _isMachineLowEndian;

int   i, j, k, l, x, y;

for (j = 0; j < *p_lin; j++) {
  y = j * ratio;
for (i = 0; i < *p_col; i++) {
  x = i * ratio;
  k = y * mszx    + x;
  l = j * *p_col + i;
  if (shouldByteSwap)
    image[l] = ((OrgImage[k] >> 8 & 0x00FF) | (OrgImage[k] << 8));
  else
    image[l] = OrgImage[k];
  }
  }
}
```

```
!==============================================================================
        function Read_Original_Images_Skip( image1, image2,
     1                              ncol, nlin, mcol, mlin,
     2                              DefFile,
     3                              PreImage, CurImage)
!==============================================================================
!       Ver. 1.0
!       Written by Akiko Kano,  Mar.24, 1993
!------------------------------------------------------------------------------
!       This function reads two image files.
!
!       (1) Image1 should correspond to current image, and Image2 should
!           correspond to previous image.
!       (2) Image matrix size is unified to the larger size between the two.
!           smaller image is compensated by filling the right and bottom part
!           with "0".
!==============================================================================
!       ARGUMENTS
!------------------------------------------------------------------------------
        implicit  none
        integer*4 Read_Original_Images_Skip
        integer*2 image1(*)             ! Current Image Data              [O]
        integer*2 image2(*)             ! Previous Image Data             [O]
        integer*4 ncol, nlin            ! Unified Matrix Size of Image Data [O]
        integer*4 mcol, mlin            ! Maximum Permitted Matrix Size    [I]
!------------------------------------------------------------------------------
!       VARIABLES
!------------------------------------------------------------------------------
        integer*4 id
        common    /LOGFILE/ id
!------------------------------------------------------------------------------
!       FUNCTIONS
!------------------------------------------------------------------------------
        integer*4      read_images_skip
!       external       read_images_skip !$pragma C(read_images_skip)
!------------------------------------------------------------------------------
!       Shige
!---------------------------------------------------------------------------
        character      DefFile*(*)
        character      PreImage*(*),     CurImage*(*)

!------------------------------------------------------------------------------
!       1. GET FILENAME OF CURRENT IMAGE - IMAGE1
!------------------------------------------------------------------------------
        Read_Original_Images_Skip = 0
        Read_Original_Images_Skip = read_images_skip(image1, image2, ncol, nlin,
     1  DefFile,
     2  PreImage,     CurImage)

!------------------------------------------------------------------------------
!               15. CREATE LOG FILE
!------------------------------------------------------------------------------
                open(id, name = 'temp.log', err = 300)
!------------------------------------------------------------------------------
!       END
!------------------------------------------------------------------------------
        return

 300    write(*,*) '!!! Log file not created. !!!'
        return
        end
```

```
/*======================================================================
===
      Subroutine read_table_
      Determination of Filename of Nonlinear Density Correction LUT
      Coded by Shige
========================================================================
===*/
#include    <stdio.h>
#include        <string.h>
#include    <stdlib.h>
#include    <time.h>

#ifdef KRL_GNU
  #define read_table read_table__
#else
  #define read_table read_table_
#endif

extern "C" void read_table(char *dcTableName, short *buf, short &ngray)
{
FILE  *fp;
short dmy;
short i;

if ((fp = fopen(dcTableName, "r") ) == NULL) {
  printf("\7\7%s can't be opened\n", dcTableName); exit(0);
  }
for (i = 0; i < ngray; i++) fscanf(fp, "%hd %hd", &dmy, buf + i);
fclose(fp);
}
```

```
/************************************************************************
****
 File Name  : ReverseValue.C
 Purpose    : flip the pixel value of an image

 Date     : 7/23/98
 Author     : Li Qiang, University of Chicago
*************************************************************************
**/

#include <iostream.h>         // C++ I/O standard stream definition
#include <fstream.h>          // file stream definition
#include "image.h"            // definition of class IMAGE
//#include "xwin.h"             // definition of functions for image
displaying
/*----------------------------------------------------------*/
/* the underbar _ following function name is                */
/* appended for Fortran                                     */
/*----------------------------------------------------------*/

#ifdef KRL_GNU
  #define reverse_value reverse_value__
#else
  #define reverse_value reverse_value_
#endif

extern "C" void reverse_value(short int *imgarr,
                         int&      col,
                         int&      row)
{
  int i;

  for (i=0 ; i<row*col ; i++)
    imgarr[i] = GRAYLEVELRANGE - imgarr[i] - 1;
}
```

```
c***********************************************************************
c
c    Purpose: To obtain ribcage points of chest image
c             Contains error solution for toplung
c
c    Limit:   This program is for chest images of any matrix size
c
c    Name:    ribcagePoint3.for
c
c    History: version 7
c
c    Required subroutine: chest_pack.for
c
c    Data:    Sept. 28, 1992
c
c    Xin-Wei Xu
c    Modified by A.Kano  Oct.8,1992
c    Modified by A.Kano  Mar.17,1993
c
c***********************************************************************

        subroutine ribcagePoint3(Image,nxw,nyh,feature,
     &                          con_No,TopPoint,Top_No,
     &                          RsidePoint,RcountSide,
     &                          LsidePoint,LcountSide)

        implicit integer*2 (i-n)

        integer*2 nxw,nyh,                ! image buffer size
     &            Image(nxw,nyh),         ! input chest image
     &            feature(50),            ! **** as discribed bellow
     &            con_No,                 ! input; No. of connect for fitting
     &            TopPoint(2,100),        ! OP; top edge points(include con_No
                                          ! edge points from each side)
     &            Top_No,                 ! OP; real top edge points number

     &            RsidePoint(2,100),RcountSide, ! OP (output)
     &            Lsidepoint(2,100),LcountSide  ! OP (output)

c ***********************************************************************

        integer*4 lungTop,                ! top position of lung
     &            imageTop                ! position of unexposed area in top
                                          ! of chest images; =1 if no such area

        integer*2 imageTop2               ! For compatibility with topRibcage
subroutine

        integer*2  RribcagePoint(2,100), ! array of ribcage points of right
                                         ! lung;(1,i) is x;(2,i) is y;
                                         ! order from midline to ribcage
     i            LribcagePoint(2,100),  ! array of ribcage point of left
                                         ! lung;
     &            R_No,L_No,             ! actual ribcage point number of both
                                         ! right and left lung
     &            RcountTop,LcountTop ! No. of top ribcage points in R & L Lung

        real*4 prof(1215),prof_smo(1215), ! profile of image & its smoothed one
     &        fd(1215),                   ! first derivative of the profile
     &        sd(1215),                   ! second derivative of the profile
     &        Rthreshold,Lthreshold,per,
     &        cf(21)
```

```
        integer*2 locationFD(1215),      ! position of fd
     &            locationSD(1215),      ! position of sd
     &            fd_No, sd_No,          ! number of fd and sd
     &            position,              ! central position of profile
     &            hwidth,                ! half width of profile

     &            tempPointX(100),tempPointY(100)

        integer*2 xcoord_hori(1000),ycoord_hori(1000),
     &            xcoord_vert(1215),ycoord_vert(1215)

        integer*2 RtopEdgein_x,RtopEdgeout_x, ! right top ribcage range
     &            RtopEdgeStart_x,RtopEdgeStart_y, ! right top rib start point
     &            LtopEdgein_x,LtopEdgeout_x, ! left top ribcage range
     &            LtopEdgeStart_x,LtopEdgeStart_y, ! left top rib start point
     &            sideEdgeStartY,RsideEdgeStartX,LsideEdgeStartX,! side start
     &            edgeUP,edgeLOW,        ! side ribcage search range
     &            hsearchX,hsearchY,     ! search range in x and y direction
     &            count,size,
     &            Rout_x,Lout_x,
     &            RM,LM,PX,PY,PXc,PYc

        integer*4 midline,kwk
        real*4      ftempPointX(100),ftempPointY(100)

c***********************************************************************
c
c               explanation of feature(50)
c
c       feature (1): lungTop    ! top of lung
c               (2): lungBottom ! bottom of lung (right diaphragm)
c               (3): vertical position of 1/5 of lung length
c               (4): vertical position of center of image
c               (5): vertical position of center of lung length
c               (6): midline position at 1/5 of lung level
c               (7): midline position at vertical center level
c               (8): midline position at center of lung level
c               (9): minimum position of right lung at 1/5 lung level (URM)
c               (10):minimum position of left lung at 1/5 lung level (ULM)
c               (11):minimum position of right lung at center lung level (LRM)
c               (12):minimum position of left lung at center lung level (LLM)
c               (13):bottom position of side ribcage fitting=
c                    lungBottom+35%*(lungBottom-lungTop)
c               (14):Right Diaphragm egde tracing X start,will be obtained from
c                    ribcagefit.for
c               (15):Right Diaphragm edge tracing X end, ontained from
c                    ribcagepoint_1k.for
c               (16):Left Diaphragm egde tracing X start, obtained from
c                    ribcagepoint_1k.for
c               (17):Left Diaphragm edge tracing X end, will be ontained from
c                    ribcagefit.for
c
c***********************************************************************     &
     nxw,nyh,           ! buffer size


c *** &&& part: initialization &&& ***

        do k=1,1215
          prof(k)=0.0
          prof_smo(k)=0.0
          fd(k)=0.0
```

```
          sd(k)=0.0
        end do

        do k=1,100
          RribcagePoint(1,k)=0
          RribcagePoint(2,k)=0
          LribcagePoint(1,k)=0
          LribcagePoint(2,k)=0
        end do

        do k=1,50
          feature(k)=0
        end do

c *** &&& end of this part &&& ***

c *** &&& part: finding top of lung &&& ***

        call top_lung_sub(Image,nxw,nyh,lungTop,imageTop,Index1)
        if ( index1.eq.0 ) then
            imageTop = 1
            lungTop = imageTop + 1
        end if
        feature(1)=lungTop
c       write(*,*) 'feature1 = ', lungTop

c *** &&& end of this part &&& ***

c *** &&& part: finding right diaphragm point, as primary lung bottom &&& ***

c**** find a hori. prof. at vert. center of image with 1/8 of image length ****

        position=int(float(nyh)/2.0+0.5)  ! vert. position of center image
        hwidth=int((float(nyh)/8.0)/2.0+0.5) ! half width of profile
        ind1=1
        ind2=1
        call profile_im_sub(Image,prof,nxw,nyh,ind1,nxw,
     &       position-hwidth,position+hwidth,ind2)       ! original profile
        inc_smo=nxw/20+1
        call prof_smo_sub(prof,nxw,nxw,inc_smo,prof_smo)   ! smoothed one

        inc_fd=nxw/20
        call fd_south_sub(prof_smo,locationFD,fd,nxw,nxw,inc_fd,fd_No) ! fd

        index=0
        call find_midline_RM_LM(prof_smo,nxw,fd,locationFD,fd_No,
     &                          midline,RM,LM,index,index,index)

        feature(4)=position
        feature(7)=midline

c***** find a vert. profile at RM with 1/12 of image width *****

        hwidth=int((float(nxw)/12.0)/2.0)
        ind1=1
        ind2=2
        call profile_im_sub(Image,prof,nxw,nyh,RM-hwidth,
     &       RM+hwidth,ind1,nyh,ind2)             ! original vert. prof.
        inc_smo=nyh/20+1
        call prof_smo_sub(prof,nyh,nyh,inc_smo,prof_smo)     ! smoothed one


        inc_fd=nyh/20
```

```
        call fd_south_sub(prof_smo,locationFD,fd,nyh,nyh,inc_fd,fd_No) ! fd

c ***** define right diaphragh at Max. fd value in lower half image ***

        do k=1,fd_No
          if (locationFD(k).ge.feature(4)) then
            k_begin=k
            go to 10
          end if
        end do
10      continue

        max_FD=locationFD(k_begin)
        fd_max=fd(k_begin)
        do i=k_begin+1,fd_No
          if (fd(i).gt.fd_max) then
            fd_max=fd(i)
            max_FD=locationFD(i)
          end if
        end do

        lungBottom=max_FD
        feature(2)=lungBottom

c *** &&& end of this part &&& ***

c *** &&& part: determine two vertical positions in the lung range &&& ***

        section=float(lungBottom-lungTop)/5.0
        level1=lungTop+int(section)       !posi. at 1/5 of lung length
        level2=lungTop+int(2.5*section)   !posi. at center of lung length
        feature(3)=level1
        feature(5)=level2

c *** &&& end of this part &&& ***

c *** &&& part: find top ribcage point search range &&& ***

c *** for right lung: RtopEdgeout_x--RtopEdgein_x
c     for left lung:  LtopEdgeout_x--LtopEdgein_x  ***

c *** obtain a hori. prof. at 1/5 lung posi. with width of 1/8 lung length ***

        hwidth=int((float(lungBottom-lungTop)/8.0)/2.0)
        ind1=1
        ind2=1
        call profile_im_sub(Image,prof,nxw,nyh,ind1,nxw,
     &       level1-hwidth,level1+hwidth,ind2)          ! original prof.
        inc_smo=nxw/20+1
        call prof_smo_sub(prof,nxw,nxw,inc_smo,prof_smo)  ! smoothed one

        inc_fd=nxw/20
        call fd_south_sub(prof_smo,locationFD,fd,nxw,nxw,inc_fd,fd_No) !fd

        index=1
        incr=nxw*3/20
        call find_midline_RM_LM(prof_smo,nxw,fd,locationFD,fd_No,
     &                          midline,RM,LM,index,feature(7),incr)

        feature(6)=midline
        feature(9)=RM
        feature(10)=LM
```

```
c *** first find inside range close to of midline, this range is defined
c     from RM or LM to midway of midline and RM or LM                  ***

        RtopEdgein_x=RM+int(float(midline-RM)/2.0)
        LtopEdgein_x=midline+int(float(LM-midline)/2.0)

c *** then find outside range between RM or LM and ribcage ***

        prof_max=prof_smo(feature(6))

        prof_RM=prof_smo(feature(9))
        prof_LM=prof_smo(feature(10))

        if (prof_RM.ge.prof_LM) then
          prof_min=prof_RM
         else
          prof_min=prof_LM
        end if

        Dif=prof_max-prof_min
        Dif2=Dif/2.0

        Rthreshold=prof_RM+Dif2
        Lthreshold=prof_LM+Dif2

        do i=RM,1,-1
          if ((prof_smo(i).le.Rthreshold).and.
     &         (prof_smo(i-1).ge.Rthreshold)) then
             RtopEdgeout_x=i-1
             go to 20
          end if
        end do
20      continue

        do i=LM,nxw
          if ((prof_smo(i).le.Lthreshold).and.
     &         (prof_smo(i+1).ge.Lthreshold)) then
             LtopEdgeout_x=i+1
             go to 30
          end if
        end do
30      continue

c *** &&& end of this part &&& ***

c *** &&& part: obtain RM and LM at level2 with 1/8 of lung length &&& ***

        ind1=1
        ind2=1
        call profile_im_sub(Image,prof,nxw,nyh,ind1,nxw,
     &                level2-hwidth,level2+hwidth,ind2) ! original profile
        inc_smo=nxw/20+1
        call prof_smo_sub(prof,nxw,nxw,inc_smo,prof_smo)        ! smoothed one

        inc_fd=nxw/20
        call fd_south_sub(prof_smo,locationFD,fd,nxw,nxw,inc_fd,fd_No)

        index=1
        incr=nxw*3/20
        call find_midline_RM_LM(prof_smo,nxw,fd,locationFD,fd_No,
     &                          midline,RM,LM,index,feature(7),incr)

        feature(8)=midline
```

```
        feature(11)=RM
        feature(12)=LM

c *** &&& end of this part &&& ***

c *** &&& part: change RM and LR of two levels by adjusting lung angles &&& ***

c *** first conside the right lung angle ***

        gy=float(feature(5)-feature(3))
        gx=float(feature(11)-feature(9))
!       angleR=atan2d(gy,gx)
        angleR=atan2(gy,gx)    ! for Linux
        angleR=180.0-angleR

        if ((angleR.lt.72.7).or.(angleR.gt.79.5)) then
          slopl=-4.041      ! set right lung angle to be 76.1 deg.
          ixCenter=int(float(feature(9)+feature(11))/2.0)
          iyCenter=int(float(feature(3)+feature(5))/2.0)

          feature(11)=ixCenter+int(float(feature(5)-iyCenter)/slopl)
          feature(9)=feature(11)-
     &               int(float(feature(5)-feature(3))/slopl)
        end if

c *** now conside left lung angles ***

        gy=float(feature(5)-feature(3))
        gx=float(feature(12)-feature(10))
!       angleL=atan2d(gy,gx)
        angleL=atan2(gy,gx)     ! for Linux
        angleL=180.0-angleL

        if ((angleL.lt.100.7).or.(angleL.gt.107.1)) then
          slopl=4.041      ! set left lung angle to be 103.9 deg.
          ixCenter=int(float(feature(10)+feature(12))/2.0)
          iyCenter=int(float(feature(3)+feature(5))/2.0)

          feature(12)=ixCenter+int(float(feature(5)-iyCenter)/slopl)
          feature(10)=feature(12)-
     &                int(float(feature(5)-feature(3))/slopl)
        end if

c *** &&& end of this part &&& ***

c*** &&& part:make straight lines upper and lower RMs or LMs,and define &&& ***
c     top ribcage start points as cross points of straight lines with
c     the horizantal line through the top of lung                      ***


        do i=1,nxw
          xcoord_hori(i)=i
          ycoord_hori(i)=feature(1)
        end do

c *** first conside right lung ***

        ind1=2
        call straight_line_sub(feature(9),feature(3),
     &       feature(11),feature(5),xcoord_vert,ycoord_vert,nyh,ind1)

        ind1=1
        ind2=2
```

```
        call cross_p_2_line_sub(xcoord_hori,ycoord_hori,nxw,ind1,
     &       xcoord_vert,ycoord_vert,nyh,ind2,PX,PY)

        RtopEdgeStart_x=PX
        RtopEdgeStart_y=PY

c *** now conside left lung ***

        ind1=2
        call straight_line_sub(feature(10),feature(3),
     &       feature(12),feature(5),xcoord_vert,ycoord_vert,nyh,ind1)

        ind1=1
        ind2=2
        call cross_p_2_line_sub(xcoord_hori,ycoord_hori,nxw,ind1,
     &       xcoord_vert,ycoord_vert,nyh,ind2,PX,PY)

        LtopEdgeStart_x=PX
        LtopEdgeStart_y=PY

c *** &&& end of this part &&& ***

c *** &&& part: set top lung position as first point of ribcage points &&& ***

        RribcagePoint(1,1)=feature(6)
        RribcagePoint(2,1)=feature(1)

        LribcagePoint(1,1)=feature(6)
        LribcagePoint(2,1)=feature(1)

c *** &&& end of this part &&& ***

c *** &&& part: trace top ribcage points &&&& ***

c ** first conside right lung top ribcage ***

c *** trace right top ribcage inside part ***

          count=1
          PX=RtopEdgeStart_x
          Py=RtopEdgeStart_y
          hsearchX=nxw/100
          hsearchY=nyh/25

100       continue

          ind1=1
          ind2=1
          ind3=1
          imageTop2 = imageTop
          call topRibcage(Image,nxw,nyh,imageTop2,PX,PY,
     &         PXc,PYc,hsearchX,hsearchY,ind1,ind2,ind3)
          imageTop = imageTop2

          count=count+1
          RribcagePoint(1,count)=PXc
          RribcagePoint(2,count)=PYc

          PX=PXc+nxw/33
          PY=PYc
          if (hsearchY.ne.nyh/33) hsearchY=nyh/33
          if (PX.le.RtopEdgein_x) go to 100
          feature(15)=PXc
```

```
c *** trace right top ribcage outside part ***

          PX=RribcagePoint(1,2)-nxw/33
          PY=RribcagePoint(2,2)
          if (PX.lt.RtopEdgeout_x) go to 300

200       continue

          ind1=1
          ind2=2
          rel_dis=float(PX-RtopEdgeout_x)/30.
          if (rel_dis.le.2.0) then
            ind3=2
           else
            ind3=1
          end if

          imageTop2 = imageTop
          call topRibcage(Image,nxw,nyh,imageTop2,PX,PY,
     &         PXc,PYc,hsearchX,hsearchY,ind1,ind2,ind3)
          imageTop = imageTop2

          count=count+1
          RribcagePoint(1,count)=PXc
          RribcagePoint(2,count)=PYc

          PX=PXc-nxw/33
          PY=PYc
          if (PX.ge.RtopEdgeout_x) go to 200

300       continue
          RcountTop=count

c *** Now conside left lung ***

c *** trace left top ribcage inside part ***

          count=1
          PX=LtopEdgeStart_x
          PY=LtopEdgeStart_y
          hsearchX=nxw/100
          hsearchY=nyh/25

400       continue

          ind1=2
          ind2=1
          ind3=1
          imageTop2 = imageTop
          call topRibcage(Image,nxw,nyh,imageTop2,PX,PY,
     &         PXc,PYc,hsearchX,hsearchY,ind1,ind2,ind3)
          imageTop = imageTop2

          count=count+1
          LribcagePoint(1,count)=PXc
          LribcagePoint(2,count)=PYc

          PX=PXc-nxw/33
          PY=PYc
          if (hsearchY.ne.nyh/33) hsearchY=nyh/33
          if (PX.ge.LtopEdgein_x) go to 400
          feature(16)=PXc
```

```
c *** trace left top ribcage outside part ***

        PX=LribcagePoint(1,2)+nxw/33
        PY=LribcagePoint(2,2)
        if (PX.gt.LtopEdgeout_x) go to 600

500     continue

        ind1=2
        ind2=2
        rel_dis=float(LtopEdgeout_x-PX)/30.
        if (rel_dis.le.2.0) then
          ind3=2
         else
          ind3=1
        end if

        imageTop2 = imageTop
        call topRibcage(Image,nxw,nyh,imageTop2,PX,PY,
     &        PXc,PYc,hsearchX,hsearchY,ind1,ind2,ind3)
        imageTop = imageTop2

        count=count+1
        LribcagePoint(1,count)=PXc
        LribcagePoint(2,count)=PYc

        PX=PXc+nxw/33
        PY=PYc
        if (PX.le.LtopEdgeout_x) go to 500

600     continue
        LcountTop=count

c *** &&& end of this part &&& ***


c *** &&& part: trace right and left side ribcage &&& ***

c *** & subpart: find side ribcage trance range and start points & ***

       LL=feature(2)-feature(1)     !! lung length in pixels
       if (LL.ge.int(float(nyh)/1.9)) then
         per=0.16
        else
         per=0.07
       end if
       feature(13)=feature(2)+int(per*float(LL))
       feature(13)=min( feature(13),(nyh-1) )

       sideEdgestartY=feature(5)   ! vertical position of center ribcage edge

       Rout_x=RribcagePoint(1,RcountTop)
       Lout_x=LribcagePoint(1,LcountTop)
                        ! Rout_x: x position of search start of right ribcage
                        ! Lout_x: x position of serach start of left ribcage

       inc_fd=nxw/40
       inc_sd=nxw/20
       call find_sideRibcageStartX(prof_smo,nxw,inc_fd,inc_sd,
     &          Rout_x,Lout_x,RsideEdgeStartX,LsideEdgeStartX)

c *** & end of this sub part& ***
```

```
c *** trace right side ribcage ***

        imageUP=1
        imageLOW=nyh

c *** trace up direction ***

        count=RcountTop
        PX=RsideEdgeStartX
        PY=sideEdgeStartY

        hsearchX=nxw/25
        hsearchY=nyh/67

1100    continue

        edgeUP=max( feature(3),(1+hsearchY) )

        ind1=1
        ind2=0
        call sideRibcage(Image,nxw,nyh,PX,PY,PXc,PYc,
     &       imageUP,imageLOW,hsearchX,hsearchY,ind1,ind2)

        if (ind2.eq.1) goto 1150
        count=count+1
        RribcagePoint(1,count)=PXc
        RribcagePoint(2,count)=PYc

        PY=PYc-nyh/33
        PX=PXc
        if (hsearchX.ne.nxw/33) hsearchX=nxw/33
        if (PY.ge.edgeUP) go to 1100


c *** trace down direction ***

1150    PX=RribcagePoint(1,RcountTop+1)
        PY=RribcagePoint(2,RcountTop+1)+nyh/33

        hsearchX=nxw/25        ! in down direction require larger search box

1200    continue

        edgeLOW=min( feature(13)+3*nyh/33,(nyh-hsearchY) )

        ind1=1
        ind2=0
        call sideRibcage(Image,nxw,nyh,PX,PY,PXc,PYc,
     &       imageUP,imageLOW,hsearchX,hsearchY,ind1,ind2)

        if (ind2.eq.1) goto 1250
        count=count+1
        RribcagePoint(1,count)=PXc
        RribcagePoint(2,count)=PYc

        PY=PYc+nyh/33
        PX=PXc
        if (PY.le.edgeLOW) go to 1200

1250    R_No=count
        RcountSide=R_No-RcountTop
```

```
c *** trace left side ribcage ***

c *** trace up direction ***

        count=LcountTop
        PX=LsideEdgeStartX
        PY=sideEdgeStartY

        hsearchX=nxw/25
        hsearchY=nyh/67

1300    continue

        edgeUP=max( feature(3),(1+hsearchY) )

        ind1=2
        ind2=0
        call sideRibcage(Image,nxw,nyh,PX,PY,PXc,PYc,
     &       imageUP,imageLOW,hsearchX,hsearchY,ind1,ind2)

        if (ind2.eq.1) goto 1350
        count=count+1
        LribcagePoint(1,count)=PXc
        LribcagePoint(2,count)=PYc

        PY=PYc-nyh/33
        PX=PXc
        if (hsearchX.ne.nxw/33) hsearchX=nxw/33
        if (PY.ge.edgeUP) go to 1300

c *** trace down direction ***

1350    PX=LribcagePoint(1,LcountTop+1)
        PY=LribcagePoint(2,LcountTop+1)+nyh/33

        hsearchX=nxw/25        ! in down direction require larger search box

1400    continue

        edgeLOW=min( feature(13)+3*nyh/33,(nyh-hsearchY) )

        ind1=2
        ind2=0
        call sideRibcage(Image,nxw,nyh,PX,PY,PXc,PYc,
     &       imageUP,imageLOW,hsearchX,hsearchY,ind1,ind2)

        if (ind2.eq.1) goto 1450
        count=count+1
        LribcagePoint(1,count)=PXc
        LribcagePoint(2,count)=PYc

        PY=PYc+nyh/33
        PX=PXc
        if (PY.le.edgeLOW) go to 1400

1450    L_No=count
        LcountSide=L_No-LcountTop

c *** &&& part: re-arrange the ribcage points &&& ***

        size=100

c *** right lung ribcage first ***
```

```
        do i=1,RcountTop
          tempPointX(i)=RribcagePoint(1,i)
          tempPointY(i)=RribcagePoint(2,i)
        end do
        call rearrange_L_S(tempPointX,tempPointY,size,RcountTop)
        do i=1,RcountTop
          RribcagePoint(1,i)=tempPointX(i)
          RribcagePoint(2,i)=tempPointY(i)
        end do

        do i=1,RcountSide
          j=RcountTop+i
          tempPointX(i)=RribcagePoint(1,j)
          tempPointY(i)=RribcagePoint(2,j)
        end do
        call rearrange_S_L(tempPointY,tempPointX,size,RcountSide)
        do i=1,RcountSide
          j=RcountTop+i
          RribcagePoint(1,j)=tempPointX(i)
          RribcagePoint(2,j)=tempPointY(i)
        end do

c *** then left lung ribcage ***

        do i=1,LcountTop
          tempPointX(i)=LribcagePoint(1,i)
          tempPointY(i)=LribcagePoint(2,i)
        end do
        call rearrange_S_L(tempPointX,tempPointY,size,LcountTop)
        do i=1,LcountTop
          LribcagePoint(1,i)=tempPointX(i)
          LribcagePoint(2,i)=tempPointY(i)
        end do

        do i=1,LcountSide
          j=LcountTop+i
          tempPointX(i)=LribcagePoint(1,j)
          tempPointY(i)=LribcagePoint(2,j)
        end do
        call rearrange_S_L(tempPointY,tempPointX,size,LcountSide)
        do i=1,LcountSide
          j=LcountTop+i
          LribcagePoint(1,j)=tempPointX(i)
          LribcagePoint(2,j)=tempPointY(i)
        end do

c *** $$$ end of this part $$$ ***

c *** &&& part: broken right and left edge points into three
c               part: TopPoint,RsidePoint and LsidePoint for
c               rib edge fitting                              &&& ***

c *** right side ***

        index=0
        iy1=feature(3)
        iy2=iy1+30
        ideg=4
        kcount=1
        kwk=0

        do i=kcount,RcountSide
```

```
          j=i+RcountTop
          ftempPointX(i)=float(RribcagePoint(1,j))
          ftempPointY(i)=float(RribcagePoint(2,j))
        end do
        kk=RcountSide-kcount+1

        ideg1 = ideg + 1
c       write(*,*)'CALLED BY RIBCAGEPOINT3.f (1)'
        call kofitc(ftempPointY,ftempPointX,kk,ideg1,cf,ideg,kwk)
        call polyfitc_integer(iy1,cf,ideg,ix1)
        call polyfitc_integer(iy2,cf,ideg,ix2)
        if (ix2.gt.ix1) then
          index=index+1
        end if

c *** left side *******

        do i=kcount,LcountSide
          j=i+LcountTop
          ftempPointX(i)=float(LribcagePoint(1,j))
          ftempPointY(i)=float(LribcagePoint(2,j))
        end do
        kk=LcountSide-kcount+1

c       write(*,*)'CALLED BY RIBCAGEPOINT3.f (2)'
        call kofitc(ftempPointY,ftempPointX,kk,ideg1,cf,ideg,kwk)
        call polyfitc_integer(iy1,cf,ideg,ix1)
        call polyfitc_integer(iy2,cf,ideg,ix2)
        if (ix2.lt.ix1) then
          index=index+1
        end if

        if (index.eq.0) then
          kcount=1
         else
          kcount=3
        end if

        kkR=RcountSide-kcount+1
        kkL=LcountSide-kcount+1
        if (kkR.le.kkL) then
          kkk=kkR
         else
          kkk=kkL
        end if

        if (index.eq.0) then
          n=con_No     ! number of ribcage edges included in TopPoint
         else
          n=con_No+5   ! number of ribcage edges included in TopPoint
          if (n.gt.kkk) n=kkk
        end if

c *****

        do i=kcount,RcountSide
          j=i+RcountTop
          l=i-kcount+1
          RsidePoint(1,l)=RribcagePoint(1,j)
          RsidePoint(2,l)=RribcagePoint(2,j)
        end do

        do i=kcount,LcountSide
```

```
          j=i+LcountTop
          l=i-kcount+1
          LsidePoint(1,l)=LribcagePoint(1,j)
          LsidePoint(2,l)=LribcagePoint(2,j)
        end do

c **********

        jcount=0

        do i=2,RcountTop ! exclude the first point which is top lung position
          jcount=jcount+1
          TopPoint(1,jcount)=RribcagePoint(1,i)
          TopPoint(2,jcount)=RribcagePoint(2,i)
        end do

        do i=1,n
          jcount=jcount+1
          TopPoint(1,jcount)=RsidePoint(1,i)
          TopPoint(2,jcount)=RsidePoint(2,i)
        end do

        do i=2,LcountTop ! exclude the first point which is top lung position
          jcount=jcount+1
          TopPoint(1,jcount)=LribcagePoint(1,i)
          TopPoint(2,jcount)=LribcagePoint(2,i)
        end do

        do i=1,n
          jcount=jcount+1
          TopPoint(1,jcount)=LsidePoint(1,i)
          TopPoint(2,jcount)=LsidePoint(2,i)
        end do

        Top_No=jcount

        do i=1,Top_No
          tempPointX(i)=TopPoint(1,i)
          tempPointY(i)=TopPoint(2,i)
        end do
        call rearrange_S_L(tempPointX,tempPointY,size,Top_No)
        do i=1,Top_No
          TopPoint(1,i)=tempPointX(i)
          TopPoint(2,i)=tempPointY(i)
        end do


c *** &&& end of this part &&& ***
        open(1, file='Rribcage.dat')
        do i = 1, 100
          write(1,1111) RribcagePoint(1,i), RribcagePoint(2,i)
        end do
        close(1)
        open(1, file='Lribcage.dat')
        do i = 1, 100
          write(1,1111) LribcagePoint(1,i), LribcagePoint(2,i)
        end do
        close(1)
 1111   format (2i5)

        return
        end
```

```
c************************************************************
c
c    program of detecting top lung ribcage by
c    using vertical search box. Currently the
c    the ribcage search is in the direction of
c    midline.
c
c    first find the two inner stop point,
c    then find the two start points
c
c    History: April 6, 92
c
c    Name: topRibcage.for
c
c*************************************************************
c
c
        subroutine topRibcage(image,nxw,nyh,top_image,xf,yf,xc,yc,
     &                          boxw_h,boxh_h,control1,control2,control3)

        implicit integer*2 (i-n)

        integer*2
     &      nxw,nyh,            ! buffer size
     &      image(nxw,nyh),     ! image buffer
     &      top_image,          !bottom of top blank white; =1 no white
     &      xf,yf,              ! x & y coordinates of former top
                                ! ribcage point
     &      xc,yc,              ! x & y coordinates of current found
                                ! top ribcage point
     &      boxw_h,boxh_h,      ! half width & half height of vert. search box
     &      bx,by,
     &      control1,           ! =1, search for right lung;
                                ! =2, search for left lung;
     &      control2,           ! =1, trace inside top edge
                                ! =2, trace outside top edge
     &      control3            ! =1, not check the level of ycp with yf
                                ! =2, do not check

        integer*2 posi(256),pp_max,pp_min,size,ycp
        real*4 prof(256),prof_smo(256),fd(256),sd(256)

        real*8 sum

c****** initialize the prof,prof_smo,fd,sd ******

        size=256

        bx=boxw_h
        by=boxh_h

        do i=1,256
          prof(i)=0.0
          prof_smo(i)=0.0
          fd(i)=0.0
          sd(i)=0.0
        end do

        xc=xf

c****** start of the program **************

        ix_start=xf-bx
```

```
        if (ix_start.lt.1) ix_start=1
        ix_end=xf+bx
        if (ix_end.gt.nxw) ix_end=nxw
        iy_start=yf-by
        if (iy_start.le.top_image) iy_start=top_image+1
        iy_end=yf+by
        if (iy_end.gt.nyh) iy_end=nyh

        NN=iy_end-iy_start+1

c****** obtain the profile in the vertical search box ******

        sum=0.0
        do iy=iy_start,iy_end
          k=iy-iy_start+1
          do ix=ix_start,ix_end
            sum=sum+float(image(ix,iy))
          end do
          sum=sum/float(ix_end-ix_start+1)
          prof(k)=sum
          sum=0.0
        end do
c****** smooth the profile **************

        inc_smo=int(float(nyh)/142.0)
        call prof_smo_sub(prof,size,NN,inc_smo,prof_smo)

c****************************************

        inc_fd=int(float(nyh)/333.0)
        call fd_south_sub(prof_smo,posi,fd,size,NN,inc_fd,No_fd)

        ii=1
        call find_minfd(posi,fd,No_fd,ii,pp_min)

          do k=pp_min,2,-1
            if ((fd(k).lt.0.0).and.(fd(k-1).gt.0.0)) then
              pp_max=posi(k)
              go to 5
            end if
          end do

          do k=pp_min,2,-1   !!!!!!!!!!!!!!!!!
            if (fd(k-1).lt.fd(k)) then
              pp_max=posi(k)
              go to 5
            end if
          end do

          k=2
          pp_max=posi(k)     !!!!!!!!!!!!!!!!!

5       continue


c*********************************************

        inc_sd=int(float(nyh)/333.0)
        call sd_south_sub(prof_smo,posi,sd,size,NN,inc_sd,No_sd)

        do k=1,No_sd
          if (posi(k).ge.pp_max) then
            k_search=k
```

```
              go to 10
            end if
          end do

10        continue

          k_min=k_search
          sd_min=sd(k_min)
          do k=k_search+1, No_sd
            if (sd(k).lt.sd_min) then
              k_min=k
              sd_min=sd(k)
            end if
          end do

c!!!!!!!!!!!!!!! new criterion !!!!!!!!!!!

          do k=k_search,k_min-1
            k1=k+1
            if ((sd(k).lt.0.0).and.(sd(k1).gt.0.0)) then
              k_mid=k
              go to 20
            end if
          end do
            ratio=0.0
            go to 21
20          continue
            k_minS=k_search
            sd_minS=sd(k_minS)
            do k=k_search+1,k_mid
              if (sd(k).lt.sd_minS) then
                sd_minS=sd(k)
                k_minS=k
              end if
            end do
            ratio=sd_minS/sd_min
21          continue

          if (ratio.gt.0.500) then ! change from 0.600 to 0.500
            ycp=iy_start+posi(k_minS)-1  !!!!
           else
            ycp=iy_start+posi(k_min)-1  !!!!
          end if

c !!!!!!!!!!!!!!!!!!!!!!!!!!!

          if (ratio.gt.0.500) then
            kps=k_minS
           else
            kps=k_min
          end if

            if ((control2.eq.2).and.(control3.eq.2)) then  !check ycp level
              if (ycp.le.yf) then
                do k=kps,No_sd-1
                  if ((sd(k).le.0.0).and.(sd(k+1).ge.0.0)) then
                    k_search=k+1
                    go to 50
                  end if
                end do

                do k=kps, No_sd-1   !!!!!!!!!!!!!!!!!
                  if (fd(k+1).lt.fd(k)) then
```

```
                    k_search=k+1
                    go to 50
                  end if
                end do

                go to 54

50              continue
                k_minT=k_search
                sd_minT=sd(k_minT)
                do k=k_search+1, No_sd
                  if (sd(k).lt.sd_minT) then
                    k_minT=k
                    sd_minT=sd(k)
                  end if
                end do
                yc=iy_start+posi(k_minT)-1  !!!!
                go to 55
               else
                yc=ycp
                go to 55
              end if
             else
              yc=ycp
              go to 55
            end if

54        yc=ycp

55        continue


          return
          end

c**************************************************
c
c   program of detecting ribcage edge using
c   horizontal search box
c
c   Name: sideRibcage.for
c
c   History: May 9,92
c
c***************************************************
c
c

          subroutine sideRibcage(image,nxw,nyh,xf,yf,xc,yc,
     &               ie_top,ie_bottom,boxw_h,boxh_h,control1,control2)


          implicit integer*2 (i-n)

          integer*2
     &        nxw,nyh,           ! buffer size
     &        image(nxw,nyh),    ! image buffer
     &        xf,yf,             ! x & y coordinates of former top
                                 ! ribcage point
     &        xc,yc,             ! x & y coordinates of current found
                                 ! top ribcage point
     &        ie_top,ie_bottom,  ! vertical range of ribcage edge
     &        boxw_h,boxh_h,     ! half width & half height of vert. search box
     &        bx,by,
```

```
     &        control1,          ! =1, search for right lung;
                                 ! =2, search for left lung;
     &        control2           ! =0, continue search
                                 ! =1, stop search

          integer*2 posi(256),pp_max,pp_min,size
          real*4 prof(256),prof_smo(256),fd(256),sd(256)

          real*8 sum

c****** initialize the prof,prof_smo,fd,sd ******

          size=256

          bx=boxw_h
          by=boxh_h

          do i=1,256
            prof(i)=0.0
            prof_smo(i)=0.0
            fd(i)=0.0
            sd(i)=0.0
          end do

          yc=yf

c****** start of the program **************

          ix_start=xf-bx
          if (ix_start.lt.1) ix_start=1
          ix_end=xf+bx
          if (ix_end.gt.nxw) ix_end=nxw
          iy_start=yf-by
          if (iy_start.le.ie_top) iy_start=ie_top
          iy_end=yf+by
          if (iy_end.gt.ie_bottom) iy_end=ie_bottom

          NN=ix_end-ix_start+1

c****** obtain the profile in the horizontal search box ******

          sum=0.0
          do ix=ix_start,ix_end
            k=ix-ix_start+1
            do iy=iy_start,iy_end
              sum=sum+float(image(ix,iy))
            end do
            sum=sum/float(iy_end-iy_start+1)
            prof(k)=sum
            sum=0.0
          end do
c****** smooth the profile **************

          inc_smo=int(float(nxw)/142.0)
          call prof_smo_sub(prof,size,NN,inc_smo,prof_smo)


c******************************************


          if (control1.eq.1) then
            inc_fd=int(float(nxw)/333.0)
            call fd_south_sub(prof_smo,posi,fd,size,NN,inc_fd,No_fd)
```

```
            call find_minfd(posi,fd,No_fd,control1,pp_min)

            do k=pp_min,2,-1
              if ((fd(k).lt.0.0).and.(fd(k-1).gt.0.0)) then
                pp_max=posi(k)
                go to 5
              end if
            end do

            do k=pp_min,2,-1   !!!!!!!!!!!!!!!!!
              if (fd(k-1).lt.fd(k)) then
                pp_max=posi(k)
                go to 5
              end if
            end do

            k=2
            pp_max=posi(k)     !!!!!!!!!!!!!!!!!

           else
            inc_fd=int(float(nxw)/333.0)
            call fd_north_sub(prof_smo,posi,fd,size,NN,inc_fd,No_fd)

            call find_minfd(posi,fd,No_fd,control1,pp_min)

            do k=pp_min, No_fd-1
              if ((fd(k).lt.0.0).and.(fd(k+1).gt.0.0)) then
                pp_max=posi(k)
                go to 5
              end if
            end do

            do k=pp_min, No_fd-1   !!!!!!!!!!!!!!!!!
              if (fd(k+1).lt.fd(k)) then
                pp_max=posi(k)
                go to 5
              end if
            end do

            k=No_fd-1
            pp_max=posi(k)     !!!!!!!!!!!!!!!!!

          end if

5         continue


c**********************************************

          inc_sd=int(float(nxw)/333.0)
          call sd_south_sub(prof_smo,posi,sd,size,NN,inc_sd,No_sd)

          do k=1,No_sd
            if (posi(k).ge.pp_max) then
              k_search=k
              go to 10
            end if
          end do

10        continue

          if (control1.eq.1) then
```

```
        k_min=k_search
        sd_min=sd(k_min)
        do k=k_search+1, No_sd
          if (sd(k).lt.sd_min) then
            k_min=k
            sd_min=sd(k)
          end if
        end do
        if (sd_min.eq.0) then
            control2=1
            return
        end if

c!!!!!!!!!!!!!!! new critien !!!!!!!!!!!

        do k=k_search,k_min-1
          k1=k+1
          if ((sd(k).lt.0.0).and.(sd(k1).gt.0.0)) then
            k_mid=k
            go to 20
          end if
        end do
          ratio=0.0
          go to 21
20        continue
          k_minS=k_search
          sd_minS=sd(k_minS)
          do k=k_search+1,k_mid
            if (sd(k).lt.sd_minS) then
              sd_minS=sd(k)
              k_minS=k
            end if
          end do
          ratio=sd_minS/sd_min
21        continue

        if (ratio.gt.0.600) then
          xc=ix_start+posi(k_minS)-1  !!!!
         else
          xc=ix_start+posi(k_min)-1  !!!!
        end if

        go to 25

c !!!!!!!!!!!!!!!!!!!!!!!!!!!!

        else

        k_min=k_search
        sd_min=sd(k_min)
        do k=k_search-1,1, -1
          if (sd(k).lt.sd_min) then
            k_min=k
            sd_min=sd(k)
          end if
        end do
        if (sd_min.eq.0) then
            control2=1
            return
        end if

c!!!!!!!!!!!!!!! new critien !!!!!!!!!!!
```

```
        do k=k_min+1, k_search-1
          k1=k+1
          if ((sd(k).gt.0.0).and.(sd(k1).lt.0.0)) then
            k_mid=k1
            go to 22
          end if
        end do
          ratio=0.0
          go to 23
22        continue
          k_minS=k_mid
          sd_minS=sd(k_minS)
          do k=k_mid+1,k_search
            if (sd(k).lt.sd_minS) then
              sd_minS=sd(k)
              k_minS=k
            end if
          end do
          ratio=sd_minS/sd_min
23        continue

        if (ratio.gt.0.600) then
          xc=ix_start+posi(k_minS)-1  !!!!
         else
          xc=ix_start+posi(k_min)-1  !!!!
        end if
        go to 25

        end if

25      continue

        return
        end
c***************************************************************************
c
c     Purpose: To find midline, right and left lung minimum PV positions
c              by analyisis of horizantal profile and its first derivative
c
c     Name: find_midline_RM_LM.for
c
c***************************************************************************

        subroutine find_midline_RM_LM(prof_smo,nxw,fd,locationFD,
     &                                fd_No,midline,RM,LM,
     &                                index,ref,increment)

        implicit integer*2 (i-n)

        integer*2 nxw,           ! x dimension of image buffer
     &            fd_No         ! actual dimension of first derivative
        integer*4
     &            midline        ! midline positino
        integer*2
     &            RM,LM,         ! minimum PV position of right & left lung
     &            locationFD(1215), ! location of each element of FD value
     &            index,         ! ?use? ref. ML;=1 use ref.ML;=0 not use
     &            ref,           ! ref.ML
     &            increment      ! search range from ref.

        real*4    prof_smo(1215),  ! smoothed profile
```

```
     &            fd(1215)         ! first derivative of profile

c***************************************************************************

c **** find midline ****

      if (index.eq.0) then

        nxq=int(float(nxw)/4.0)
        max_PV=nxq
        prof_max=prof_smo(max_PV)
        do i=nxq+1, 3*nxq
          if (prof_smo(i).gt.prof_max) then
            prof_max=prof_smo(i)
            max_PV=i
          end if
        end do
        midline=max_PV   ! find midline within range of: nxq--3*nxp

      end if

      if (index.eq.1) then

        istart=ref-increment
        iend=ref+increment
        max_PV=istart
        prof_max=prof_smo(max_PV)
        do i=istart+1, iend
          if (prof_smo(i).gt.prof_max) then
            prof_max=prof_smo(i)
            max_PV=i
          end if
        end do
        midline=max_PV   ! find midline within range of: istart--iend

      end if

        per90=0.9*prof_max
        do i=max_PV,1,-1
          if ((prof_smo(i).ge.per90).and.(prof_smo(i-1).le.per90)) then
            i_per90_r=i
            go to 10
          end if
        end do
10      continue

        do i=max_PV,nxw
          if ((prof_smo(i).ge.per90).and.(prof_smo(i+1).le.per90)) then
            i_per90_l=i
            go to 20
          end if
        end do
20      continue

c **** find RM position ****

        do k=1,fd_No
          if (locationFD(k).ge.i_per90_r) then
            k_r=k
            go to 30
          end if
        end do
```

```
30      continue
        do k=1,fd_No
          if (locationFD(k).ge.i_per90_l) then
            k_l=k
            go to 40
          end if
        end do
40      continue

        do i=k_r, 1, -1
          i1=i-1
          if ((fd(i).gt.0.0).and.(fd(i1).lt.0.0)) then
            aveRM=float(locationFD(i)+locationFD(i1))/2.0
            RM=int(aveRM)
            go to 50
          end if
        end do
50      continue

c **** find  LM position ****


        do i=k_l, fd_No
          i1=i+1
          if ((fd(i).lt.0.0).and.(fd(i1).gt.0.0)) then
            aveLM=float(locationFD(i)+locationFD(i1))/2.0
            LM=int(aveLM)
            go to 60
          end if
        end do
60      continue

        return
        end
c***************************************************************************
c
c      Purpose: use fd and sd of hori. profile at vertical center of lung
c               length to determine side ribcage start points(only x positions)
c
c      Name: sideRibcageStart_x.for
c
c***************************************************************************


        subroutine find_sideRibcageStartX(prof,nxw,inc_fd,inc_sd,
     &             Rout_x,Lout_x,RsideEdgeStartX,LsideEdgeStartX)

        implicit integer*2 (i-n)

        integer*2 nxw,     ! image width
     &            inc_fd,inc_sd, ! increment for FD and SD
     &            Rout_x,Lout_x, ! positions where side ribcage start points
                                 ! located outside them
     &            RsideEdgeStartX,LsideEdgeStartX  ! start points( x posi.)
                                                   ! of right and left side rib
        real*4 prof(1215)

c***************************************************************************

        integer*2 fd_No,sd_No,
     &            locationFD(1215),locationSD(1215)

        integer*2 RsideStartX_fd,RsideStartX_sd,
     &             LsideStartX_fd,LsideStartX_sd

        real*4 fd(1215),sd(1215)

        real*4 RFDposi,LFDposi,RSDposi,LSDposi

c *** conside right lung side ribcage start point x posi. first ***


        ifd=inc_fd
        isd=inc_sd
        ind=0

1       call fd_south_sub(prof,locationFD,fd,nxw,nxw,ifd,fd_No)
        call sd_south_sub(prof,locationSD,sd,nxw,nxw,isd,sd_No)


        do i=1, fd_No
          if (locationFD(i).ge.Rout_x) then
            kRend=i
            go to 10
          end if
        end do
10      continue

        do i=kRend,2,-1
          i1=i-1
          if ((fd(i).lt.0.0).and.(fd(i1).gt.0.0)) then
            Index=0
            RFDposi=float(locationFD(i)+locationFD(i1))/2.0
            RsideStartX_fd=int(RFDposi)
            go to 20
          end if
        end do
        Index=1
20      continue

        do i=1, sd_No
          if (locationSD(i).ge.Rout_x) then
            kRend=i
            go to 30
          end if
        end do
30      continue

        do i=kRend,2,-1
          i1=i-1
          if ((sd(i).gt.0.0).and.(sd(i1).lt.0.0)) then
            RSDposi=float(locationSD(i)+locationSD(i1))/2.0
            RsideStartX_sd=int(RSDposi)
            go to 37
          end if
        end do
        ind=ind+1
        if (ind.lt.2) then
          isd=ifd
          go to 1
        end if

37      continue

         if (Index.eq.0) then
          if (abs(RsideStartX_sd-RsideStartX_fd).gt.nxw/10) go to 40
```

```
          RsideEdgeStartX=int(float(RsideStartX_fd+RsideStartX_sd)/2.0)
          go to 50
        end if

40      continue

        sd_min=sd(kRend)
        min=kRend
        do j=kRend-1,1,-1
          if (sd(j).lt.sd_min) then
            sd_min=sd(j)
            min=j
          end if
        end do

        if (min.eq.1) then
          RsideEdgeStartX=int(float(RsideStartX_sd+locationSD(min))/2.0)
         else
          RsideEdgeStartX=locationSD(min)
        end if

50      continue

c *** conside left lung side ribcage start point x posi. now ***

        do i=1, fd_No
          if (locationFD(i).ge.Lout_x) then
            kLstart=i
            go to 60
          end if
        end do
60      continue

        do i=kLstart, fd_No-1
          i1=i+1
          if ((fd(i).gt.0.0).and.(fd(i1).lt.0.0)) then
            Index=0
            LFDposi=float(locationFD(i)+locationFD(i1))/2.0
            LsideStartX_fd=int(LFDposi)
            go to 70
          end if
        end do
        Index=1
70      continue


        do i=1, sd_No
          if (locationSD(i).ge.Lout_x) then
            kLstart=i
            go to 80
          end if
        end do
80      continue

        do i=kLstart, sd_No-1
          i1=i+1
          if ((sd(i).gt.0.0).and.(sd(i1).lt.0.0)) then
            LSDposi=float(locationSD(i)+locationSD(i1))/2.0
            LsideStartX_sd=int(LSDposi)
            go to 87
          end if
        end do
```

```
        ind=ind+1
        if (ind.lt.2) then
          isd=ifd
          go to 1
        end if


87      continue

        if (Index.eq.0) then
          if (abs(LsideStartX_sd-LsideStartX_fd).gt.nxw/10) go to 90

          LsideEdgeStartX=int(float(LsideStartX_fd+LsideStartX_sd)/2.0)
          go to 100
        end if

90      continue

        sd_min=sd(kLstart)
        min=kLstart
        do j=kLstart+1, sd_No
          if (sd(j).lt.sd_min) then
            sd_min=sd(j)
            min=j
          end if
        end do
        if (min.eq.sd_No) then
          LsideEdgeStartX=int(float(LsideStartX_sd+locationSD(min))/2.0)
         else
          LsideEdgeStartX=locationSD(min)
        end if

100     continue

        return
        end
c************************************************************
c
c subroutine: to fine min. position in first derivative
c
c       name find_minfd(posi,fd,No_fd,pp_min,contl)
c
c************************************************************

        subroutine find_minfd(posi,fd,No_fd,contl,pp_min)

        implicit integer*2 (i-n)

        integer*2 posi(256),pp_min,No_fd,range_No,index,
     $          contl !=1, top or right ribcage;=2 left ribcage

        real*4 fd(256),ave,
     &      range(4,20) ! j=1...20, can have 20 ranges;
                        ! for j, i=1,2,3,4: Head,Tail,midfd,posi_minfd


c*******************************************************************

        range_No=0

        index=0
        do i=2,No_fd-2
          if (fd(i).gt.0.0) index=index+1
```

```
        end do
        if (index.eq.0) then
          ave=0.0
          do i=2,No_fd-2
            ave=ave+fd(i)
          end do
          ave=ave/float(No_fd-2-2+1)
         else
          ave=0.0
        end if

        do i=1,No_fd-1
          if ((fd(i).gt.ave).and.(fd(i+1).lt.ave)) then
            range_No=range_No+1
            range(2,range_No)=float(i)
          end if
        end do

        range_No=range_No+1
        range(2,range_No)=float(No_fd-1)

        range(1,1)=2.0
        do j=2,range_No
          range(1,j)=range(2,j-1)+1
        end do

        do j=1,range_No
          posi_min=range(1,j)
          fd_min=fd(int(range(1,j)))
          do k=int(range(1,j)+1),int(range(2,j))
            if (fd(k).lt.fd_min) then
              posi_min=float(k)
              fd_min=fd(k)
            end if
          end do
          range(3,j)=fd_min
          range(4,j)=posi_min
        end do

        fdF=range(3,1)
        j_indF=1
        do j=2,range_No
          if (range(3,j).lt.fdF) then
            fdF=range(3,j)
            j_indF=j
          end if
        end do

        if (contl.eq.1) then
          if (j_indF.eq.1) then
            pp_min=int(range(4,1))
            ratio=0.0
            go to 100
          end if
         else
          if (j_indF.eq.range_No) then
            pp_min=int(range(4,range_No))
            ratio=0.0
            go to 100
          end if
        end if

          j_indS=1
        fdS=range(3,1)
        do j=2,range_No
          if (range(3,j).lt.fdS) then
            if (j.eq.j_indF) go to 10
            fdS=range(3,j)
            j_indS=j
          end if
10        continue
        end do

      if (contl.eq.1) then
        if (j_indS.gt.j_indF) then
          pp_min=int(range(4,j_indF))
          ratio=0.0
          go to 100
        end if
       else
        if (j_indS.lt.j_indF) then
          pp_min=int(range(4,j_indF))
          ratio=0.0
          go to 100
        end if
      end if

      ratio=range(3,j_indS)/range(3,j_indF)
      if (ratio.gt.0.500) then
        pp_min=int(range(4,j_indS))
       else
        pp_min=int(range(4,j_indF))
      end if

100   continue

      return
      end
```

```
c***************************************************************************c
c       This subroutine is to fit the detected ribcage and top lung edges
c       into smooth lung bounday. The boundary curve is seperated into 2
c       sets, i.e. right and left sets.
c
c++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

        subroutine Ribcage_Detection( image, ncol, nlin, feature,
     1   Rribcage, Rribcage_No, Lribcage, Lribcage_No,cf_r,cf_l)

c===========================================================================
c       Ver. 3.0
c       Written by Xin-Wei Xu,  "RIBCAGEFIT_NEW2"
c       Modified by Akiko Kano,  Mar.24, 1993
c       Modified by Xin-Wei Xu,  Apr.22, 1993
c       Modified by Xin-Wei Xu,  12/10/93
c----------------------------------------------------------------------------
c       This subroutine detects ribcage edges of a chest image and returns the
c       X,Y-locations of the ribcage points and the number of the points.
c
c       (1) The ribcage curve is separated into right lung side and left lung
c           side, about a midline at 1/5 top lung level.
c       (2) If a detected ribcage curve is concave in the top lung,
c           the top part is ignored. The curve starts with the highest
c           point in its side and goes down.
c       (3) Input image should have grayscale ranging from 0 to 1023, in which
c           "0" corresponds to high optical density and "1023" corresponds to
c           low optical density.
c----------------------------------------------------------------------------
c       This subroutine calls;
c               ribcagepoint3, kofitc_pack, chest_pack3
c============================================================================
c       ARGUMENTS
c----------------------------------------------------------------------------
        implicit integer*2 (i-n)

        integer*4 ncol, nlin            ! Matrix Size of Image Data        [I]
        integer*2 image(ncol,nlin)      ! Input Chest Image Data           [I]
        integer*2 feature(50)           ! Features of Chest Image          [O]
        integer*2 Rribcage(2,1215)      ! Right Lung Side Ribcage Points   [O]
                                        ! Rribcage(1,*):X
                                        ! Rribcage(2,*):Y -> Increases 1 by 1
        integer*2 Rribcage_No           ! No. of Right Side Ribcage Points  [O]
        integer*2 Lribcage(2,1215)      ! Left Lung Side Ribcage Points    [O]
                                        ! Lribcage(1,*):X
                                        ! Lribcage(2,*):Y -> Increases 1 by 1
        integer*2 Lribcage_No           ! No. of Left Side Ribcage Points   [O]
c----------------------------------------------------------------------------
c       VARIABLES
c----------------------------------------------------------------------------
        integer*2 nxw,nyh,
     &            TopFit(2,1215),TopFit_No, ! fitted points of top ribcage
     &            RsideFitNew(2,1215),RsideFitNew_No,! fit. pot. of R. side rib
     &            LsideFitNew(2,1215),LsideFitNew_No ! fit. pot. of L. side rib
        integer*2 TopPoint(2,100),Top_No,
     &            RsidePoint(2,100),RcountSide,
     &            LsidePoint(2,100),LcountSide,
     $            con_No,
     $            RconX1,RconX2,LconX1,LconX2,conY1,conY2,
     $            RconL,RconU,LconL,LconU,
     &            deg,deg1,row,collum,count,
     &            RsideFit(2,1215),RsideFit_No,
     &            LsideFit(2,1215),LsideFit_No,
```

```
     %            tempX(1215),tempY(1215),tempX2(1215),tempY2(1215),
     %            dy
        integer   rc0,rc1,lc0,lc1
        real*4    cf_r(21), cf_l(21)!ribcage fitting coef. of R. & L. lungs
        real      tempPointX(100),tempPointY(100),
     &            cf(21)    ! polynomial cofficients
c----------------------------------------------------------------------------

c----------------------------------------------------------------------------
c       DESCRIPTION OF FEATURES
c----------------------------------------------------------------------------
c       feature (1): lungTop    ! top of lung
c               (2): lungBottom ! bottom of lung (right diaphragm)
c               (3): vertical position of 1/5 of lung length
c               (4): vertical position of center of image
c               (5): vertical position of center of lung length
c               (6): midline position at 1/5 of lung level
c               (7): midline position at vertical center level
c               (8): midline position at center of lung level
c               (9): minimum position of right lung at 1/5 lung level (URM)
c               (10):minimum position of left lung at 1/5 lung level (ULM)
c               (11):minimum position of right lung at center lung level (LRM)
c               (12):minimum position of left lung at center lung level (LLM)
c               (13):bottom position of side ribcage fitting=
c                    lungBottom+certain % of (lungBottom-lungTop)
c               (14):Right Diaphragm egde tracing X start, will be obtained
c                    from ribcage_detection.for
c               (15):Right Diaphragm edge tracing X end, ontained from
c                    ribcagepoint3.for
c               (16):Left Diaphragm egde tracing X start, obtained from
c                    ribcagepoint3.for
c               (17):Left Diaphragm edge tracing X end, will be ontained from
c                    ribcage_detection.for
c----------------------------------------------------------------------------
c       BEGIN
c----------------------------------------------------------------------------

        nxw = ncol
        nyh = nlin

        do i=1,50
          feature(i)=0
        end do

        do i=1,1215
          Rribcage(1,i)=0
          Rribcage(2,i)=0
          Lribcage(1,i)=0
          Lribcage(2,i)=0
        end do

        con_No=3

10      call ribcagePoint3(image,nxw,nyh,feature,
     &                     con_No,TopPoint,Top_No,
     &                     RsidePoint,RcountSide,
     &                     LsidePoint,LcountSide)

c**************************************************************************

c *** $$$ find connect range $$$ ***

        do i=1,10
```

```
          cf_r(i)=0.0
          cf_l(i)=0.0
        end do

        n=con_No      ! connect points in side ribcage

        maxX=TopPoint(1,1)
        minX=TopPoint(1,1)
        do i=2, n
          if (TopPoint(1,i).gt.maxX) maxX=TopPoint(1,i)
          if (TopPoint(1,i).lt.minX) minX=TopPoint(1,i)
        end do
        RconX1=minX
        RconX2=maxX

        maxX=TopPoint(1,Top_No)
        minX=TopPoint(1,Top_No)
        do i=Top_No-1,Top_No-n+1, -1
          if (TopPoint(1,i).gt.maxX) maxX=TopPoint(1,i)
          if (TopPoint(1,i).lt.minX) minX=TopPoint(1,i)
        end do
        LconX1=minX
        LconX2=maxX

        conY1=RsidePoint(2,1)
        conY2=RsidePoint(2,n)

c *** &&& end of this part &&& ***

c *** $$$ part: curve fitting part &&& ***

c *** top lung first ***

        deg=4 ! using 4th order polynomial fuction in top ribcage
        deg1=deg+1

        do i=1, Top_No
          tempPointX(i)=float(TopPoint(1,i))
          tempPointY(i)=float(TopPoint(2,i))
        end do

        ind=0
        call kofitc(tempPointX,tempPointY,Top_No,
     &              deg1,cf,deg,ind)
        !   fitting relation: Y=a0+a1*X+a2*X**2+a3*X**3

        count=0
        do collum=RconX1,LconX2
          ix=collum
          call polyfitc_integer(ix,cf,deg,iy)
          count=count+1
          TopFit(1,count)=ix
          TopFit(2,count)=iy
        end do                  ! fitting the right ribcage
        TopFit_No=count

c *** right lung second ***

        deg=4 ! using 4th order polynomial functions for side ribcages
        deg1=deg+1

        do i=1,RcountSide
          tempPointX(i)=float(RsidePoint(1,i))
```

```
          tempPointY(i)=float(RsidePoint(2,i))
        end do

        ind=0
        call kofitc(tempPointY,tempPointX,RcountSide,
     &              deg1,cf,deg,ind)
        !   fitting relation: X=a0+a1*Y+a2*Y**2+a3*Y**3

        do i=1,deg1
          cf_r(i)=cf(i)
        end do

        count=0
        do row=conY1,feature(13)
          iy=row
          call polyfitc_integer(iy,cf,deg,ix)
          count=count+1
          RsideFit(1,count)=ix
          RsideFit(2,count)=iy
        end do                  ! fitting the right ribcage
        RsideFit_No=count

c *** then left ribcage ***

        do i=1,LcountSide
          tempPointX(i)=float(LsidePoint(1,i))
          tempPointY(i)=float(LsidePoint(2,i))
        end do

        ind=0
        call kofitc(tempPointY,tempPointX,LcountSide,
     &              deg1,cf,deg,ind)
        !   fitting relation: X=a0+a1*Y+a2*Y**2+a3*Y**3

        do i=1,deg1
          cf_l(i)=cf(i)
        end do

        count=0
        do row=conY1,feature(13)
          iy=row
          call polyfitc_integer(iy,cf,deg,ix)
          count=count+1
          LsideFit(1,count)=ix
          LsideFit(2,count)=iy
        end do                  ! fitting the left ribcage
        LsideFit_No=count

c *** &&& end of this part &&& ***

c *** &&& make connection and modify fitted ribcage curves &&& ***

c *** top part right ***

        RconL=TopFit(2,1)
        RconU=conY1

        do i=1,TopFit_No
          i1=i+1
          if ((TopFit(2,i).ge.RconU).and.(TopFit(2,i1).le.RconU)) then
            nU=i
            go to 500
          end if
```

```
      end do
500   continue

      a=1.0/float(nU-1+1)
      k=0
      do i=1,nU
        k=k+1
        ak=a*float(k)
        bk=1.0-ak
        kref=TopFit(2,i)
        do j=1,RsideFit_No
          if (RsideFit(2,j).eq.kref) then
            TopFit(1,i)=int(ak*float(TopFit(1,i))+
     &                      bk*float(RsideFit(1,j)))
            go to 505
          end if
        end do
505     continue
      end do

c *** top part left ***

      LconL=TopFit(2,TopFit_No)
      LconU=conY1

      do i=TopFit_No,1,-1
        i1=i-1
        if ((TopFit(2,i).ge.LconU).and.(TopFit(2,i1).le.LconU)) then
          nU=i
          go to 600
        end if
      end do
600   continue

      a=1.0/float(TopFit_No-nU+1)
      k=0
      do i=TopFit_No,nU,-1
        k=k+1
        ak=a*float(k)
        bk=1.0-ak
        kref=TopFit(2,i)
        do j=1,LsideFit_No
          if (LsideFit(2,j).eq.kref) then
            TopFit(1,i)=int(ak*float(TopFit(1,i))+
     &                      bk*float(LsideFit(1,j)))
            go to 605
         end if
         end do
605      continue
      end do

c *** right side ***

      do i=1,RsideFit_No
        if (RsideFit(2,i).eq.RconL) then
          js=i+1
          go to 700
        end if
      end do
700   continue

      RsideFitNew_No=RsideFit_No-js+1
      do i=1,RsideFitNew_No
```

```
        j=i+js-1
        RsideFitNew(1,i)=RsideFit(1,j)
        RsideFitNew(2,i)=RsideFit(2,j)
      end do

c *** left side ***

      do i=1,LsideFit_No
        if (LsideFit(2,i).eq.LconL) then
          js=i+1
          go to 800
        end if
      end do
800   continue

      LsideFitNew_No=LsideFit_No-js+1
      do i=1,LsideFitNew_No
        j=i+js-1
        LsideFitNew(1,i)=LsideFit(1,j)
        LsideFitNew(2,i)=LsideFit(2,j)
      end do

c *** &&& end of this part &&& ***

c *** reformat the ribcage into two parts: right and left ribcage curve ***

      ML=feature(6)
      i=1
      do while (TopFit(1,i).lt.ML)
        i=i+1
      end do
      jML=i

c *** first reformat right side ribcage curve ***

      do i=1,1215
        tempX(i)=0
        tempY(i)=0
        tempX2(i)=0
        tempY2(i)=0
      end do

      i=1
      do while ( i.lt.jML .and. TopFit(2,i).ge.TopFit(2,i+1) )
        i=i+1                              ! process right half top ribcage curve
      end do
      jMLr=i

      do i=1,jMLr
        tempX(i)=TopFit(1,i)
        tempY(i)=TopFit(2,i)
      end do

c *** part to eliminate data which y position is same ***

      do i=jMLr,2,-1
        j=i-1
        do k=j,1,-1
          if (tempY(k).eq.tempY(i)) then
            tempY(k)=0
          end if
        end do
      end do
```

```
      rc0=0
      do i=jMLr,1,-1
        if (tempY(i).ne.0) then
          rc0=rc0+1
          tempX2(rc0)=tempX(i)
          tempY2(rc0)=tempY(i)
        end if
      end do

c *** end of this part ***

c *** part of inserting y value ***

      rc1=0
      do i=1,rc0-1
        rc1=rc1+1
        Rribcage(1,rc1)=tempX2(i)
        Rribcage(2,rc1)=tempY2(i)
        i1=i+1
        dy=tempY2(i1)-tempY2(i)
        if (dy.gt.1) then
          iy=tempY2(i)
          iy1=tempY2(i1)
          do j=iy+1,iy1-1
            rc1=rc1+1
            Rribcage(1,rc1)=tempX2(i)
            Rribcage(2,rc1)=j
          end do
        end if
      end do

c *** end of this part ***

c *** make up whole right side ribcage curve by adding right side edges ***

      do i=1,RsideFitNew_No
        rc1=rc1+1
        Rribcage(1,rc1)=RsideFitNew(1,i)
        Rribcage(2,rc1)=RsideFitNew(2,i)
      end do
      Rribcage_No=rc1

c *** end of this part ***

      feature(14)=Rribcage(1,Rribcage_No-5) ! !!!

c &&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

c *** now reformat left side ribcage curve ***

      do i=1,1215
        tempX(i)=0
        tempY(i)=0
        tempX2(i)=0
        tempY2(i)=0
      end do

      tempX(1)=TopFit(1,1)
      tempY(1)=TopFit(2,1)
      i=TopFit_No
      do while ( i.gt.jML .and. TopFit(2,i).ge.TopFit(2,i-1) )
        i=i-1                              ! process left half top ribcage curve
```

```
      end do
      jMLl=i

      do i=jMLl,TopFit_No ! process left half top ribcage curve
        tempX(i)=TopFit(1,i)
        tempY(i)=TopFit(2,i)
      end do

c *** part to eliminate data which y position is same ***

      do i=jMLl,TopFit_No-1
        j=i+1
        do k=j,TopFit_No
          if (tempY(k).eq.tempY(i)) then
            tempY(k)=0
          end if
        end do
      end do

      lc0=0
      do i=jMLl,TopFit_No
        if (tempY(i).ne.0) then
          lc0=lc0+1
          tempX2(lc0)=tempX(i)
          tempY2(lc0)=tempY(i)
        end if
      end do

c *** end of this part ***

c *** part of inserting y value ***

      lc1=0
      do i=1,lc0-1
        lc1=lc1+1
        Lribcage(1,lc1)=tempX2(i)
        Lribcage(2,lc1)=tempY2(i)
        i1=i+1
        dy=tempY2(i1)-tempY2(i)
        if (dy.gt.1) then
          iy=tempY2(i)
          iy1=tempY2(i1)
          do j=iy+1,iy1-1
            lc1=lc1+1
            Lribcage(1,lc1)=tempX2(i)
            Lribcage(2,lc1)=j
          end do
        end if
      end do

c *** end of this part ***

c *** make up whole left side ribcage curve by adding right side edges ***

      do i=1,LsideFitNew_No
        lc1=lc1+1
        Lribcage(1,lc1)=LsideFitNew(1,i)
        Lribcage(2,lc1)=LsideFitNew(2,i)
      end do
      Lribcage_No=lc1

c *** end of this part ***
```

```
        feature(17)=Lribcage(1,Lribcage_No-5) ! !!!!!

!--------------------------------------------------------------------------------
!       END
!--------------------------------------------------------------------------------
        return
        end
```

```
!=========================================================================
        function ROI_Selection( image2, ncol, nlin,
     &             feature1, rribcage1, lribcage1,
     &             feature2, rribcage2, lribcage2, rribcage_no2, lribcage_no2,
     &             sas, tps, inc, ldlimit, number, sac, tpc, region1, region2 ,
     &             shiftmid, ribtop)
!=========================================================================
!       Ver. 1.0
!       Written by Akiko Kano,  Mar.24, 1993
!-------------------------------------------------------------------------
!       This function selects centers of template ROIs on Image2 and correspond-
!       ing search area ROIs on Image1 for local matchings.
!
!       (1) Template ROIs are selected in the lung area including the neck on
!           Image2.
!       (2) Search area ROIs are then located on Image1 by shifting all the
!           template ROIs.   Global shift is performed for the right lung area
!           and for the left lung area, independently.   The amount of shift is
!           determined based on the x-location of the midline at 1/5 of top
!           lung level and the y-location of the highest point on the ribcage
!           curve.
!       (3) Template ROI is not selected if average pixel value of the center
!           and surrounding 8 pixels is less than (ld_limit).
!       (4) In the following cases, quit processing and returns "error".
!           a. Top lung is lower than 30 % line from the top of the image.
!           b. Midline is apart from the center more than 15 % of the image
!              width.
!           c. Shift value y for the right lung area and the left lung area are
!              different from each other by more than 4 % of the image height.
!=========================================================================
!       ARGUMENTS
!-------------------------------------------------------------------------
        implicit  none
        integer*4 ROI_Selection
        integer*4 ncol, nlin                ! Matrix Size of Image Data          [I]
        integer*2 image2(ncol,nlin)         ! Original Image Data 2              [I]
        integer*2 feature1(50)              ! Features of Chest Image 1          [I]
        integer*2 rribcage1(2,1215)         ! Right Lung Side Ribcage Points     [I]
                                            ! Rribcage(1,*):X
                                            ! Rribcage(2,*):Y -> Increases 1 by 1
        integer*2 lribcage1(2,1215)         ! Left Lung Side Ribcage Points      [I]
                                            ! Lribcage(1,*):X
                                            ! Lribcage(2,*):Y -> Increases 1 by 1
        integer*2 feature2(50)              ! Features of Chest Image 2          [I]
        integer*2 rribcage2(2,1215)         ! Right Lung Side Ribcage Points     [I]
                                            ! Rribcage(1,*):X
                                            ! Rribcage(2,*):Y -> Increases 1 by 1
        integer*2 rribcage_No2              ! No. of Right Side Ribcage Points   [I]
        integer*2 lribcage2(2,1215)         ! Left Lung Side Ribcage Points      [I]
                                            ! Lribcage(1,*):X
                                            ! Lribcage(2,*):Y -> Increases 1 by 1
        integer*2 lribcage_No2              ! No. of Left Side Ribcage Points    [I]
        integer*4 sas                       ! Search Area ROI Size (Pixels)      [I]
        integer*4 tps                       ! Template ROI Size (Pixels)         [I]
        integer*4 inc                       ! Distance Between ROIs (Pixels)     [I]
        integer*4 ldlimit                   ! Pixel Value for Low Density Limit of
                                            !  ROI Selection                     [I]
        integer*4 number                    ! Number of Template ROIs            [O]
        integer*4 sac(2,*)                  ! Locations of Search Area Centers   [O]
                                            ! sac(1,*):X, sac(2,*):Y
        integer*4 tpc(2,*)                  ! Locations of Template Centers      [O]
                                            ! tpc(1,*):X, tpc(2,*):Y
        integer*4 region1(4),region2(4)     ! Rectangle Region Including All Points
```

```
                                            ! region*(1,2):Upper-Left            [O]
                                            ! region*(3,4):Lower-Right
!-------------------------------------------------------------------------
!       VARIABLES
!-------------------------------------------------------------------------
        real*4    P_TOP, P_MID, P_DIF
c       parameter (P_TOP=0.3, P_MID=0.15, P_DIF=0.04)
        parameter (P_TOP=0.4, P_MID=0.15, P_DIF=0.04)
c       character str*80
        integer*4 rdiff(2), ldiff(2)
        integer*4 mid, rtop, ltop, top, bottom
        integer*4 rim1, rim2, rim3, neck, pave
        integer*4 MC, TL, ML, C, L, CC, LL
        integer*4 SC, EC, SL, EL, OSC, OEC
        integer*4 NR, NL
        integer*4 rlimit, llimit, tlimit, blimit
        integer*4 chtop, chmid, chdif
        integer*4 id
c       integer*4 leng
        integer*4 shiftmid, ribtop
        common    /LOGFILE/ id
!-------------------------------------------------------------------------
!       FUNCTIONS
!-------------------------------------------------------------------------
c       integer*4 Find_SideColumn ! This routine must be defined as subroutine.
c       integer*4 PutOutputF, UTL$STR_PRINT, UTL$FILE_WRITE
!-------------------------------------------------------------------------
!
!-------------------------------------------------------------------------
!       1. DETERMINE SHIFT VALUES
!-------------------------------------------------------------------------
c       rdiff(1) = feature1(6) - feature2(6)
c       ldiff(1) = feature1(6) - feature2(6)
c       rdiff(2) = rribcage1(2,1) - rribcage2(2,1)
c       ldiff(2) = lribcage1(2,1) - lribcage2(2,1)
c       rdiff(2) = feature1(3) - feature2(3)
c       ldiff(2) = feature1(3) - feature2(3)
        rdiff(1)=shiftmid
        ldiff(1)=shiftmid
        rdiff(2)=ribtop
        ldiff(2)=ribtop
c       write(*,*) 'rdiff(1) = ', rdiff(1), '  ldiff(1) = ', ldiff(1)
c       write(*,*) 'rdiff(2) = ', rdiff(2), '  ldiff(2) = ', ldiff(2)
!-------------------------------------------------------------------------
!       2. DISPLAY PARAMETERS
!-------------------------------------------------------------------------
c       leng = UTL$STR_PRINT
c       1        ( str, ' Template Size=%i  Search Area Size=%i  Distance=%i',
c       2                                                     tps, sas, inc )
c       call UTL$FILE_WRITE( id, str )
c       leng = UTL$STR_PRINT( str, ' Low Density Limit=%i', ldlimit )
c       call UTL$FILE_WRITE( id, str )

        write(id,*) ' Template Size = ', tps, ' Search Area Size = ', sas,
     1                          ' Distance = ',inc
        write(id,*) ' Low Density Limit = ', ldlimit

c       call PutOutputF( ' ' )
c       write(*,*) ' '
        mid = feature1(6)
        rtop = rribcage1(2,1)
        ltop = lribcage1(2,1)
        top = feature1(1)
```

```
        bottom = feature1(2)
c       leng = UTL$STR_PRINT( str,
c       1    ' Image1 : Mid(1/5)=%3i  RTop=%3i  LTop=%3i  Top=%3i  Bottom=%3i',
c       2                                        mid, rtop, ltop, top, bottom )
c       call PutOutputF( '%s', str )
c       call UTL$FILE_WRITE( id, str )

        write(id,1000) ' Image1 : Mid(1/5) = ', mid, ' Rtop = ', rtop, ' Ltop = ',
     1                      ltop, ' Top = ', top, ' Bottom = ', bottom
 1000   format(A, I3, A, I3, A, I3, A, I3, A, I3)
        mid = feature2(6)
        rtop = rribcage2(2,1)
        ltop = lribcage2(2,1)
        top = feature2(1)
        bottom = feature2(2)
c       leng = UTL$STR_PRINT( str,
c       1    ' Image2 : Mid(1/5)=%3i  RTop=%3i  LTop=%3i  Top=%3i  Bottom=%3i',
c       2                                        mid, rtop, ltop, top, bottom )
c       call PutOutputF( '%s', str )
c       call UTL$FILE_WRITE( id, str )

        write(id,1000) ' Image2 : Mid(1/5) = ', mid, ' Rtop = ', rtop, ' Ltop = ',
     1                      ltop, ' Top = ', top, ' Bottom = ', bottom
c       write(*,1000) ' Image2 : Mid(1/5) = ', mid, ' Rtop = ', rtop, ' Ltop = ',
c     1                     ltop, ' Top = ', top, ' Bottom = ', bottom

c       leng = UTL$STR_PRINT( str,' Shift  : Right(%3i,%3i)',rdiff(1),rdiff(2))
c       call PutOutputF( '%s', str )
c       call UTL$FILE_WRITE( id, str )

        write(id,1010) rdiff(1), rdiff(2)
 1010   format( ' Shift : Right(', I3, ',', I3, ')')

c       leng = UTL$STR_PRINT( str,' Shift  : Left (%3i,%3i)',ldiff(1),ldiff(2))
c       call PutOutputF( '%s', str )
c       call UTL$FILE_WRITE( id, str )

        write(id,1020) ldiff(1), ldiff(2)
 1020   format( ' Shift : Left (',I3, ',',I3, ')')
!-------------------------------------------------------------------------
!       3. CHECK TOP AND MIDLINE
!-------------------------------------------------------------------------
        ROI_Selection = 0

        chtop = nint( P_TOP * float(nlin) )
        chmid = nint( P_MID * float(ncol) )
        chdif = nint( P_DIF * float(nlin) )

        if ( rribcage1(2,1).gt.chtop .or. lribcage1(2,1).gt.chtop ) then
c           call PutOutputF
c       1        ( '!!! Top lung of Image1 is too low, may be incorrect. !!!' )
            write(*,*) '!!! Top lung of Image1 is too low, may be incorrect. !!!'
            return
        end if
        if ( rribcage2(2,1).gt.chtop .or. lribcage2(2,1).gt.chtop ) then
c           call PutOutputF
c       1        ( '!!! Top lung of Image2 is too low, may be incorrect. !!!' )
            write(*,*) '!!! Top lung of Image2 is too low, may be incorrect. !!!'
            return
        end if

        if ( abs(feature1(6)-ncol/2).gt.chmid ) then
c           call PutOutputF
```

```
c       1        ( '!!! Midline of Image1 is one-sided, may be incorrect. !!!' )
            write(*,*) '!!! Midline of Image1 is one-sided, may be incorrect. !!!'
            return
        end if
        if ( abs(feature2(6)-ncol/2).gt.chmid ) then
c           call PutOutputF
c       1        ( '!!! Midline of Image2 is one-sided, may be incorrect. !!!' )
            write(*,*) '!!! Midline of Image2 is one-sided, may be incorrect. !!!'
            return
        end if

        if ( abs(rdiff(2)-ldiff(2)).gt.chdif ) then
c           call PutOutputF
c       1 ('!!! Right and left top lung are unbalanced, may be incorrect. !!!')
            write(*,*) '!!! Right and left top lung are unbalanced. May be
incorrect. !!!'
            return
        end if
!-------------------------------------------------------------------------
!       4. DETERMINE OUTSIDE LIMIT
!-------------------------------------------------------------------------
        rlimit = max( 1+tps/2, 1-rdiff(1) )
        llimit = min( ncol-tps/2, ncol-ldiff(1) )
        tlimit = max( 1+tps/2, max( 1-rdiff(2), 1-ldiff(2) ) )
        blimit = min( nlin-tps/2, min( nlin-rdiff(2), nlin-ldiff(2) ) )
!-------------------------------------------------------------------------
!       5. DETERMINE HOW MUCH BIGGER THE ROI SELECTION AREA IS THAN RIBCAGE AREA
!-------------------------------------------------------------------------
        rim1 = inc * 2
        rim2 = tps / 2
        rim3 = tps / 4
        neck = inc * 2
!-------------------------------------------------------------------------
!       6. DETERMINE TOP LUNG(TP), 1/5 FROM TOP(ML), AND MIDLINE(MC)
!-------------------------------------------------------------------------
        TL = min( rribcage2(2,1), lribcage2(2,1) )
        ML = feature2(3)
        MC = feature2(6)
!-------------------------------------------------------------------------
!       7. DETERMINE START LINE AND END LINE
!-------------------------------------------------------------------------
        SL = TL - rim1
        SL = max( SL, tlimit )
        EL = max( rribcage2(2,rribcage_no2),lribcage2(2,lribcage_no2) ) + inc
        EL = min( EL, blimit )
!-------------------------------------------------------------------------
!       8. INITIALIZE
!-------------------------------------------------------------------------
        L = SL
        NR = 1
        NL = 1
        OSC = MC - neck
        OEC = MC + neck
        number = 0
        region2(1) = OSC
        region2(3) = OEC
!-------------------------------------------------------------------------
!       9. DETERMINE START COLUMN AND END COLUMN FOR EACH LINE
!-------------------------------------------------------------------------
        do while ( L.le.EL )
!-------------------------------------------------------------------------
!           9-1. NECK ( HIGHER THAN (TL-INC) )
!-------------------------------------------------------------------------
```

```
          if ( L.lt.(TL-inc) ) then
              SC = OSC
              EC = OEC
!-----------------------------------------------------------------------
!         9-2. TOP LUNG ( HIGHER THAN ML )
!-----------------------------------------------------------------------
          else if ( L.le.ML ) then
              if ( L .lt. rribcage2(2,1)-rim2 ) then
                  SC = min( rribcage2(1,1), OSC )
              else if (  L .le. rribcage2(2,1) ) then
                  call Find_SideColumn( L+rim2, SC, OSC, rribcage2, NR, -rim2,
     1                                  rlimit )
              else
                  call Find_SideColumn( L, SC, OSC, rribcage2, NR, -rim2,
     1                                  rlimit )
              end if

              if ( L .lt. lribcage2(2,1)-rim2 ) then
                  EC = max( lribcage2(1,1), OEC )
              else if (  L .le. lribcage2(2,1) ) then
                  call Find_SideColumn( L+rim2, EC, OEC, lribcage2, NL, rim2,
     1                                  llimit )
              else
                  call Find_SideColumn( L, EC, OEC, lribcage2, NL, rim2,
     1                                  llimit )
              end if

              OSC = SC
              OEC = EC
!-----------------------------------------------------------------------
!         9-3. MID AND BOTTOM LUNG ( LOWER THAN ML )
!-----------------------------------------------------------------------
          else
              if ( L.le.rribcage2(2,rribcage_no2) ) call Find_SideColumn
     1                         ( L, SC, OSC, rribcage2, NR, -rim3, rlimit )
              if ( L.le.lribcage2(2,lribcage_no2) ) call Find_SideColumn
     1                         ( L, EC, OEC, lribcage2, NL, rim3, llimit )
          end if
!-----------------------------------------------------------------------
!         10. SELECT CENTER OF TEMPLATE
!-----------------------------------------------------------------------
          C = MC
!-----------------------------------------------------------------------
!         10-1. START SEARCH
!-----------------------------------------------------------------------
          do while ( C.ge.SC+inc )
              C = C - inc
          end do
          if ( C.lt.region2(1) ) region2(1) = C
          do while ( C.le.EC )
!-----------------------------------------------------------------------
!             10-2. CHECK PIXEL VALUE
!-----------------------------------------------------------------------
              pave = 0
              do LL = L-1, L+1
                  do CC = C-1, C+1
                      pave = pave + image2(CC,LL)
                  end do
              end do
              pave = pave / 9
              if ( pave.le.ldlimit ) then
!-----------------------------------------------------------------------
!                 10-3. DETERMINE CENTER
```

```
!-----------------------------------------------------------------------
                  number = number + 1
                  tpc(1,number) = C
                  tpc(2,number) = L
!-----------------------------------------------------------------------
!                 11. SELECT CENTER OF CORRESPONDING SEARCH AREA
!-----------------------------------------------------------------------
                  if ( C.le.MC ) then
                      sac(1,number) = C + rdiff(1)
                      sac(2,number) = L + rdiff(2)
                  else
                      sac(1,number) = C + ldiff(1)
                      sac(2,number) = L + ldiff(2)
                  end if
              end if
              C = C + inc
c              write(*,*)number,    tpc(1,number), tpc(2,number)
          end do
          if ( (C-inc).gt.region2(3) ) region2(3) = C - inc
          L = L + inc
      end do
!-----------------------------------------------------------------------
!     12. DISPLAY TOTAL NUMBER OF POINTS
!-----------------------------------------------------------------------
c     call PutOutputF( ' Total %i Points.', number )
c     call PutOutputF( ' ' )
c     leng = UTL$STR_PRINT( str, ' Number of ROI Pairs : %i', number )
c     call UTL$FILE_WRITE( id, str )

c     write(*,*) ' Total ', number, ' Points.'
c     write(*,*) ' '

      write(id,*) ' Number of ROI Pairs : ', number

      if ( number.gt.0 ) ROI_Selection = 1
!-----------------------------------------------------------------------
!     13. DETERMINE REGION1 AND REGION2
!-----------------------------------------------------------------------
      region2(2) = SL
      region2(4) = L - inc
      region1(1) = region2(1) + rdiff(1)
      region1(2) = region2(2) + min( rdiff(2),ldiff(2) )
      region1(3) = region2(3) + ldiff(1)
      region1(4) = region2(4) + max( rdiff(2),ldiff(2) )
!-----------------------------------------------------------------------
!     END
!-----------------------------------------------------------------------
      return
      end
!
!=======================================================================
      subroutine Find_SideColumn( L, C, OC, ribcage, N, rim, limit )
!=======================================================================
!     Ver. 1.0
!     Written by Akiko Kano,  Dec.24, 1992
!-----------------------------------------------------------------------
!     This function determines the start column and the end column on a given
!     line, for selection of centers of ROIs.
!
!     (1) Points are selected inside ribcage curves and in borders outside of
!         the curves.
!     (2) The width of the borders is defined by "rim".   If "rim" is a
!         positive value, it means search of an end column (left lung).
```

```
!         If "rim" is a negative value, it means search of a start column
!         (right lung).
!     (3) Points are not selected outer than "OC".
!=======================================================================
!     ARGUMENTS
!-----------------------------------------------------------------------
      implicit  none
c     integer*4 Find_SideColumn
      integer*4 L                    ! Line No.                        [I]
      integer*4 C                    ! Found Column No.                [O]
      integer*4 OC                   ! Outside Limit on This Line      [I]
      integer*2 ribcage(2,1215)      ! Ribcage Points                  [I]
                                     !  ribcage(1,*):X, ribcage(2,*):Y
      integer*4 N                    ! Point No. of ribcage          [I,O]
      integer*4 rim                  ! Width of Border Outside Rib-Cage [I]
      integer*4 limit                ! Outside Limit on This Image     [I]
!-----------------------------------------------------------------------
!     BEGIN
!-----------------------------------------------------------------------
      do while ( ribcage(2,N).lt.L )
          N = N + 1
      end do

      if ( abs(rim) .eq. rim ) then
          C = min( max( (ribcage(1,N)+rim), OC ), limit )
      else
          C = max( min( (ribcage(1,N)+rim), OC ), limit )
      end if
!-----------------------------------------------------------------------
!     END
!-----------------------------------------------------------------------
      return
      end
```

```
/************************************************************************
****
 File Name  : RotAngleByRibcage.C
 Purpose    : Detecting lateral inclination angle by using ribcage
method

 Method     : first, we get the middle points of left and right ribcage,
            then fit a line to the points, and take the line as the
midline

 Date     : 5/1/98
 Author     : Li Qiang, University of Chicago
************************************************************************
**/

#include <iostream.h>         // C++ I/O standard stream definition
#include <fstream.h>          // file stream definition
#include <math.h>             // Math definition
#include "image.h"            // definition of class IMAGE
//#include "xwin.h"             // definition of functions for image
displaying

/*-----------------------------------------------------------*/
/* the underbar _ following rot_angle_by_ribcage is         */
/* appended for Fortran                                      */
/* you should call this function in Fortran by               */
/*  call rot_angle_by_ribcage(...)                           */
/* Ugly? as a C programmer, YES. However, observe the rule  */
/*-----------------------------------------------------------*/

#ifdef KRL_GNU
  #define least_square_1d least_square_1d__
  #define rot_angle_by_ribcage rot_angle_by_ribcage__
#else
  #define least_square_1d least_square_1d_
  #define rot_angle_by_ribcage rot_angle_by_ribcage_
#endif

extern "C" void   least_square_1d(int     &N,        // a subroutine by
Kano in Fortran
                         float   *X,
                         float   *Y,
                         float   &A,
                         float   &B,
                         float   *error,
                         float   &sd_err );


extern "C" void rot_angle_by_ribcage(short     *imgarr,         //
midline is overlapped to this array
                                                         // used only
for debug
                              int&      ncol,            // array size
must be 586*586
                              int&      nlin,
                              short     *RibFeature,
                              short     rribcage1[1215][2],
                              short&    rribcage_no1,
                              short     lribcage1[1215][2],
                              short&    lribcage_no1,
                              float&    A,                     // OUT:
```

```
coeffi. of y=A*x+B
                              float&    B,
                              float&    BestAngle,              // OUT:
angle of midline
                              float&    BestShift)              // OUT:
shift of midline

// at Y coor. of (nlin/2)
{
  int i;

  //-------------------------------------
  //            overlapping ribcage, and
  //            its medial axis
  //-------------------------------------

  for (i=0 ; i<rribcage_no1 ; i++) {
    imgarr[rribcage1[i][0] + ncol * rribcage1[i][1]] = 0;
    imgarr[lribcage1[i][0] + ncol * lribcage1[i][1]] = 0;
  }

  #define MAXNO 1000
  float X[MAXNO], Y[MAXNO], error[MAXNO];
  float    sd_err;
  int mid_no = 0;

  // get a set of middle points of ribcage
  for ( i = rribcage_no1/7 ; i < rribcage_no1-rribcage_no1/10 ; i += 5)
{
    X[mid_no] = (lribcage1[i][0]+rribcage1[i][0])/2;
    Y[mid_no] = rribcage1[i][1];
    mid_no = mid_no + 1;
    //    imgarr[(lribcage1[i][0]+rribcage1[i][0])/2 + ncol * rribcage1
[i][1]] = 0;
    //    imgarr[(lribcage1[i+1][0]+rribcage1[i+1][0])/2 + ncol *
rribcage1[i+1][1]] = 0;
    //    imgarr[(lribcage1[i+2][0]+rribcage1[i+2][0])/2 + ncol *
rribcage1[i+2][1]] = 0;
    //    imgarr[(lribcage1[i+3][0]+rribcage1[i+3][0])/2 + ncol *
rribcage1[i+3][1]] = 0;
    //    imgarr[(lribcage1[i+4][0]+rribcage1[i+4][0])/2 + ncol *
rribcage1[i+4][1]] = 0;
  }

  // fit the points to a line
  least_square_1d( mid_no, Y, X, A, B, error, sd_err );

  // usually, no need to check contents of 'error' and 'sd_err'
  BestAngle = -atan(A)*180.0/M_PI ;
  BestShift = (1.0*ncol/2) - ((nlin/2)*A+B);

  mid_no = 0;
  for ( i=rribcage_no1/7 ; i < rribcage_no1-rribcage_no1/10 ; i++ ) {
    float j=rribcage1[i][1]*A+B;
    X[mid_no] = j;
    Y[mid_no] = rribcage1[i][1];
    imgarr[(int) (X[mid_no] + ncol * Y[mid_no])]=0;
    imgarr[(int) (X[mid_no] + 1 + ncol * Y[mid_no])]=0;
    mid_no = mid_no + 1;
  }
  cerr << "MIDLINE: Angle = " << BestAngle << "     Shift = " <<
BestShift << endl;
  //  cout << X[mid_no-1] << " " << Y[mid_no-1] << " " << X[mid_no-1]-X
[0] << " " <<
  //    Y[mid_no-1]-Y[0] << " " << BestAngle << endl;


}
```

```
/***********************************************************************
****
 File Name  : rotate.c
 Purpose    : rotate an image. Image center is the rotattion center
 Date     : Feb. 23,1998
 Author     : Li Qiang, University of Chicago
***********************************************************************
**/
/*----------------------------------------------------------------*/
/* the underbar _ following function name is used for Fortran   */
/* you should call this function in Fortran by                  */
/*        call image_rotate(...)                                */
/* and call it in C by                                          */
/*        image_rotate__(...);                                  */
/* Ugly? as a C programmer, YES. However, observe the rule      */
/*----------------------------------------------------------------*/

#include <stdio.h>
#include <stdlib.h>
//#include <iostream.h>
#include <math.h>

#ifdef KRL_GNU
  #define image_rotate image_rotate__
#else
  #define image_rotate image_rotate_
#endif

extern "C" void image_rotate(short *img,    /* image data, saved in a 1-
D array */
                       int& col,      /* image size
*/
                       int& row,
                       float& angle,  /* rotation angle
*/
                       int& blank)    /* value for background, usually 0
*/
{
  register int i, j;
  float TmpX, TmpY;
  int IntX, IntY;
  float DeltaX, DeltaY;
  float SinV, CosV;
  short *img1;
  float x0, y0;   /* center of rotation  */

  x0 = col /2.0;
  y0 = row /2.0;

  /*--------------------------------------------------------------*/
  /* allocate a new array to contain the result                 */
  /*--------------------------------------------------------------*/
  img1 = new short[col * row];

  CosV= cos( angle * M_PI/180.0);
  SinV= sin( angle * M_PI/180.0);

  /*--------------------------------------------------------------*/
  /* rotation is performed in a backward transform, i.e.,       */
  /* for each pixel in the ROTATED image, we calculate its      */
  /* corresponding pixel in the original image. if any, the     */
  /* corresponding pixel is assigned to the pixel in the ROTATED*/
  /* image. if the backward transformed pixel is out of the     */
  /* original image, assign blank to that pixel                */
  /* In this way, rotation can be performed very EASILY         */
  /*------------------------------------------------------------*/

  for (i= 0; i< row; i++)
  for (j= 0; j< col; j++) {
    TmpX = x0 + CosV*(j-x0) - SinV*(i-y0) ;
    TmpY = y0 + SinV*(j-x0) + CosV*(i-y0) ;
    IntX = (int) TmpX;
    IntY = (int) TmpY;
    DeltaX = TmpX - (float) IntX;
    DeltaY = TmpY - (float) IntY;
    if (IntX >= 0 && IntX < col-1 && IntY >= 0 && IntY < row-1 ) {
      //      img1[i * col + j] =  img[IntY * col + IntX];
      img1[i * col + j] =  (int) (img[IntY * col + IntX] * (1.0-DeltaY)
* (1.0-DeltaX) +
                                img[(IntY+1) * col + IntX] * DeltaY *
(1.0-DeltaX) +
                                img[IntY * col + IntX + 1] * (1.0-
DeltaY) * DeltaX +
                                img[(IntY+1) * col + IntX + 1] * DeltaY
* DeltaX + 0.5);
    }
    else
      img1[i * col + j] = blank;
  }

  for (i= 0; i< row; i++)
  for (j= 0; j< col; j++)
    img[i * col + j] = img1[i * col + j];

  delete img1;
}
```

```
!==============================================================================
      subroutine Save_DataQ( number, sac, tpc, inc, region2, ncol, nlin,
     1                     DX, DY, CC, FX, FY, file1, file2, leng1, leng2,
     2                     sas, tps, order )
!==============================================================================
!     Ver. 1.0
!     Written by Akiko Kano,  Apr.8, 1993
!------------------------------------------------------------------------------
!     This function saves shift maps for pair of DX and DY (initial shift
!     values) or FX and FY (fitted shift values) as a text file.
!
!     (1) The file format is;
!
!          X1    Y1    X2    Y2    SHIFT_X    SHIFT_Y    [CC]
!
!         X1 : Column No. for the center of search area ROI on Image1
!         Y1 : Line   No. for the center of search area ROI on Image1
!         X2 : Column No. for the center of template ROI on Image2
!         Y2 : Line   No. for the center of template ROI on Image2
!         SHIFT_X : Initial or fitted shift value in X direction for (X2,Y2)
!         SHIFT_Y : Initial or fitted shift value in Y direction for (X2,Y2)
!         CC : Cross-correlation value (only for initial shift values)
!
!         In a file for fitted shift values, X1 and Y1 are "0" if the point
!         (X2,Y2) was not used as a center of a template in local matching.
!
!     (2) Filenames will be;
!         (FILE1)-(FILE2)_T(TPSIZE)S(SASIZE)I(DIS)_Q.DAT          : Initial
!         (FILE1)-(FILE2)_T(TPSIZE)S(SASIZE)I(DIS)_F(ORDER)_Q.DAT : Fitted
!         Example : CTS001B-CTS001A_T32S64I16_Q.DAT
!                   CTS001B-CTS001A_T32S64I16_F10_Q.DAT
!==============================================================================
!     ARGUMENTS
!------------------------------------------------------------------------------
      implicit  none
c     integer*4 Save_DataQ
      integer*4 number               ! Number of ROI Pairs                [I]
      integer*4 sac(2,*)             ! Centers of Search Areas on Image1  [I]
      integer*4 tpc(2,*)             ! Centers of Templates on Image2     [I]
      integer*4 inc                  ! Distant Between Centers of ROIs    [I]
      integer*4 region2(4)           ! Smallest Rect. Area Including tpc  [I]
      integer*4 ncol, nlin           ! Matrix Size of Image Data          [I]
      integer*4 DX(*), DY(*)         ! Initial Shift Values               [I]
      real*4    CC(*)                ! Cross-Correlation Values           [I]
      real*4    FX(ncol,nlin),FY(ncol,nlin)   ! Fitted Shift Values       [I]
      character file1*(*), file2*(*)  ! Filenames of Image1 and Image2    [I]
      integer*4 leng1, leng2         ! Length of Filenames                [I]
      integer*4 sas                  ! Search Area ROI Size (Pixels)      [I]
      integer*4 tps                  ! Template ROI Size (Pixels)         [I]
      integer*4 order                ! Order of Polynomials for Fitting   [I]
!------------------------------------------------------------------------------
!     VARIABLES
!------------------------------------------------------------------------------
      character name1*80, name2*80
      integer*4 id1, id2, nameleng1, nameleng2, status
      integer*4 X, Y, C, L, I
      integer*4 id
      common    /LOGFILE/ id
!-----------------------------------------------------------------------------
!     Shige
!-----------------------------------------------------------------------------
!     external        fname1_printq !$pragma C(fname1_printq)
!     external        fname2_printq !$pragma C(fname2_printq)
```

```
!------------------------------------------------------------------------------
!     FUNCTIONS
!------------------------------------------------------------------------------
c     integer*4 UTL$FILE_CREATE, UTL$FILE_CLOSE, UTL$FILE_WRITE,UTL$STR_PRINT
!------------------------------------------------------------------------------
!
!------------------------------------------------------------------------------
!     FORMATS
!------------------------------------------------------------------------------
 100  format ( 6i6, 3x, f8.5 )
 200  format ( 4i6, 2f9.2 )
!------------------------------------------------------------------------------
!     1. DETERMINE FILENAMES
!------------------------------------------------------------------------------

      call fname1_printq(file1, file2, name1, tps, sas, inc)
      call fname2_printq(file1, file2, name2, tps, sas, inc, order)
!------------------------------------------------------------------------------
!     2. CREATE FILE(1)
!------------------------------------------------------------------------------
      id1 = 7
      open(id1, name = name1, err = 300)
      write(*,*) '... Creating Data File', name1
      write(id,*) ' Data File 1       :'//name1
!------------------------------------------------------------------------------
!     3. CREATE FILE(2)
!------------------------------------------------------------------------------
      id2 = 8
      open(id2, name = name2, err = 400)
      write(*,*) '... Creating Data File ', name2
      write(id,*) ' Data File 2       :'//name2
!------------------------------------------------------------------------------
!     4. WRITE DATA(1)
!------------------------------------------------------------------------------
      do I = 1, number
          write ( id1, 100 )
     1         sac(1,I),sac(2,I), tpc(1,I), tpc(2,I), DX(I), DY(I), CC(I)
      end do
!------------------------------------------------------------------------------
!     5. WRITE DATA(2)
!------------------------------------------------------------------------------
      I = 1
      do L = region2(2), region2(4), inc
          do C = region2(1), region2(3), inc
              if ( C.eq.tpc(1,I) .and. L.eq.tpc(2,I) ) then
                  X = sac(1,I)
                  Y = sac(2,I)
                  I = I + 1
              else
                  X = 0
                  Y = 0
              end if
              write ( id2, 200 ) X, Y, C, L, FX(C,L), FY(C,L)
          end do
      end do
!------------------------------------------------------------------------------
!     6. CLOSE FILES
!------------------------------------------------------------------------------
c     call UTL$FILE_CLOSE( id1 )
c     call UTL$FILE_CLOSE( id2 )
c     Save_DataQ = 1
      close( id1 )
      close( id2 )
```

```
!------------------------------------------------------------------------------
!     END
!------------------------------------------------------------------------------
      return
 300  continue
      write(*,*) '!!! Data File ', name1, ' not created. !!!'
      return
 400  continue
      write(*,*) '!!! Data File ', name2, ' not created. !!!'
      return
      end
```

SAVE_SUBTRACTION_IMAGEQ.f

```
!==============================================================================
        subroutine Save_Subtraction_ImageQ(image, ncol, nlin,
        2                                  DefFile,
        3                                  PreImage,    CurImage,
        4                                  SubImage)
!==============================================================================
!       Ver. 1.0
!       Written by Akiko Kano,  Apr.8, 1993
!------------------------------------------------------------------------------
!       This function saves a image.
!==============================================================================
!       ARGUMENTS
!------------------------------------------------------------------------------
        implicit  none
        integer*4 ncol, nlin              ! Matrix Size of Image Data         [I]
        integer*2 image(ncol,nlin)        ! Image Data                        [I]
        character DefFile*(*)             ! Definition File                   [I]
        character PreImage*(*)            ! Previous Image Name               [I]
        character CurImage*(*)            ! Current Image Name                [I]
        character SubImage*(*)            ! Subtracted Image Name             [O]
!------------------------------------------------------------------------------
!       VARIABLES
!------------------------------------------------------------------------------
!       external        write_imageq !$pragma C(write_imageq)
!------------------------------------------------------------------------------
!       BEGIN
!------------------------------------------------------------------------------

        call write_imageq(image, ncol, nlin ,
     1                    DefFile,
     2                    PreImage,     CurImage,     SubImage)

!------------------------------------------------------------------------------
!       END
!------------------------------------------------------------------------------
        return
        end
```

```
/***********************************************************************
****
 File Name  : shift.c
 Purpose    : shift an image in x direction.
 Date     : May 18,1998
 Author     : Li Qiang, University of Chicago
************************************************************************
**/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#ifdef KRL_GNU
  #define image_shift image_shift__
#else
  #define image_shift image_shift_
#endif


extern "C" void image_shift(short *img,    /* image data, saved in a 1-D
array */
                      int *col,      /* image size
*/
                      int *row,
                      float *shift,  /* shift value in X direction
*/
                      int *blank)    /* value for background, usually 0
*/
{
  register int i, j;
  int TmpX, TmpY;
  short *img1;

  /*------------------------------------------------------------*/
  /* allocate a new array to contain the result                 */
  /*------------------------------------------------------------*/
  img1 = new short[*col * *row];

  for (i= 0; i< *row; i++)
  for (j= 0; j< *col; j++) {
    TmpY = i ;
    TmpX = j - (int) (*shift+0.5) ;
    if (TmpX > 0 && TmpX < *col)
      img1[i * *col + j] = img[TmpY * *col + TmpX];
    else
      img1[i * *col + j] = *blank;
  }

  for (i= 0; i< *row; i++)
  for (j= 0; j< *col; j++)
    img[i * *col + j] = img1[i * *col + j];

  delete img1;
}
```

```
!==========================================================================
        function Shift_Map_Fitting_Intp(DX,DY,CC,FX,FY,ncol,nlin,inc,
        1   number,tpc,sac,region2, WF, order,FITX,FITY,weight,
        2   npx,npy,tpco,saco,IDX,IDY,CCo,angle,shiftmid,ribtop,
        3   iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
        4   ix1_rule3,ith_avgpix,image1,giveangle,midline,
        5   iflag,iterative,toplung,MAXPT,IDX2,IDY2,anglewk,roiavg,
        6   roisd,histo)
!==========================================================================
!       Ver. 1.0
!       Written by Akiko Kano,  Apr.9, 1993
!       Modified by Taka Ishida, Jan.5, 1998
!       Add comments by Taka Ishida, Jan.9, 1998
!       Bug fixed (Reported by Shige) Taka Ishida, Jan.29, 1999
!  (Replaced tpco by saco in lines
352,360,361,369,370,371,415,470,551,585,598,611,624)
!--------------------------------------------------------------------------
!       This function performs a weighted two-dimensional curve fitting with
!       N-th order polynomials on shift values DX and DY for the first warping
!       i.e., when iterative = 1.
!       For the second warping, shift values DX and DY are determined by
!       linear interpolation, i.e., when iterative = 2.
!
!       (1) Using detected ribcage edges and cardiac edges, lung areas are
!           segmentated for determination of weights for surface fitting.
!           If the center location of the ROI is in the lung area, the weight
!           will be 1.00.
!           If the center location of the ROI is not in the lung area, the weight
!           will be 0.25.  (see  parameter weightlow)
!           Note important modification: The weights are determined according to
!           location of ROIs, nor from LUT of weighting factors vs cross-correlation
!           value.
!
!       (2) Make accumurated shift vector orientation histogram for determination
!           of dominant vector.
!           If the maximum peak of the histogram is larger than two times of the
!           average value of the histogram, the shift vectors within peak plus
!           minus 90 degree become dominant vector.
!           Otherwise, 0-180 degree will be considered as dominant shift vector.
!
!       (3) Fitting algorithm is based on "TRN2DN" by S. Katsuragawa.  Fittings
!           are applied only for the center points of templates.  Shift values
!           for other points inbetween are calculated by linear interpolations.
!           The surface fitting technique is applied for only first wraping.
!           Dominant shift vector or non-dominant shift vector will be used for
!           the surfacce fitting.
!           If a parameter of giveangle = 1, dominant shift vector will used for
!           the surface fitting.
!           If a parameter of giveangle = -1, non-dominant shift vector will used
!           for the surface fitting.
!
!       (4) For the second warping, shift vactor are determined by linear
!           interporation.
!
!
!-------------------       <Discription of flag conbinations>
--------------------------
!       First, this subroutine was called from the main subroutine two times
!       for determination of two fitted shift-values.
!
!       One is for determination of fitted dominant shift-vector, the other is
!       for determination of fitted non-dominant shift-vector.
!
```

```
!       Thus, we create two subtraction images.
!
!       Then, we compare with the histogram widths of two subtraction images
!       in each lung.
!
!       If the histogram widths of subtraction image in both right and left lungs
!       obtained by dominant shift-vector is lower than another, iflag will be 0.
!       If the histogram widths of subtraction image in both right and left lungs
!       obtained by non-dominant shift-vector is lower than another, iflag will be
-1.
!       If the histogram widths of subtraction image in only left lungs
!       obtained by dominant shift-vector is lower than another, iflag will be 1.
!       If the histogram widths of subtraction image in only right lungs
!       obtained by dominant shift-vector is lower than another, iflag will be 2.
!
!       In case, iflag is 1 or 2, this subroutine was called from the main
subroutine
!       again for determination of flipped shift-values.
!       Otherwise, goto 2nd warping.
!
!   <1st call>
! iterative  giveangle   iflag
!    1           1         0    :   1st warping, Dominant shift-vector
!    1          -1         0    :   1st warping, Non-dominant shift-vector
!
!   <2nd call>
!    1           1         1    :   1st warping, Dominant shift-vector    , Flip
vectors in R lung
!    1           1         2    :   1st warping, Dominant shift-vector    , Flip
vectors in L lung
!    1          -1         1    :   1st warping, Non-dominant shift-vector, Flip
vectors in R lung
!    1          -1         2    :   1st warping, Non-dominant shift-vector, Flip
vectors in L lung
!
!   <3rd call>
!    2           1         0    :   2nd warping, Final shift-vectors are determined
by using
!                                                linear interporation of all shift
vectors.
! Vector flip switch
! iflag =  0 -> Use dominant orientation
! for fitting.
! iflag = -1 -> Use not dominant orientation
! for fitting.
! iflag =  1 -> Switch vector for R lung
! iflag =  2 -> Switch vector for L lung
! Loop counter of warping iteration [I]
!--------------------------------------------------------------------------
!       This function calls;    Weighted_Fit_Intp
!==========================================================================
!       ARGUMENTS
!--------------------------------------------------------------------------
        implicit  none
        integer*4 Shift_Map_Fitting_Intp
        integer*4 DX(*), DY(*)              ! Initial Shift Values            [I]
        real*4    CC(*)                     ! Cross-Correlation Values        [I]
        integer*4 ncol, nlin                ! Matrix Size of Image Data       [I]
        real*4    FX(ncol,nlin)             ! Matrix of Fitted Shift Value DX [O]
        real*4    FY(ncol,nlin)             ! Matrix of Fitted Shift Value DY [O]
        integer*4 FITX(*), FITY(*)          ! Fitted Shift Values             [O]
        integer*4 inc                       ! Distance Increment of ROIs(Pixels)[I]
        integer*4 number                    ! Number of Templates             [I]
```

```
        integer*4 tpc(2,*)                  ! Centers of Templates on Image2    [I]
        integer*4 sac(2,*)                  ! Centers of SearchROI on Image1    [I]
        integer*4 region2(4)                ! Smallest Rect. Area Including tpc [I]
        real*4    WF(2,11)                  ! LUT of Weighting Factor vs. Cross-
                                            ! Correlation Value                 [I]
                                            ! Not used in this version.!!!!!!
        integer*4 order                     ! Order of Polynomials for Fitting  [I]
        integer*4 shiftmid,ribtop           ! Global shift of X and Y           [I]
!--------------------------------------------------------------------------
        integer*4 ix1_rule2,iy1_rule2       ! Upper-right location of mediastinum [I]
        integer*4 ix2_rule2,iy2_rule2       ! Upper-left of left cardiac edge     [I]
        integer*4 ix1_rule3(2,nlin)          ! x-locations of cardiac edges       [I]
                                            ! ix1_rule3(1,nlin) => right cardiac
x-location
                                            ! ix1_rule3(2,nlin) => left cardiac
x-location
        integer*4 iy1_rule1                 ! bottom of right cardiac edge        [I]
                                            ! cardiac edges are determined by
                                            ! chs_sub.f developed by Taka.
!
!       (ix1_rule2,iy1_rule2) --------> x        |
!                                       |        |
!                                       |        |
!                                       |        |
!                                       |        x <-------- (ix2_rule2,iy2_rule2)
!                                      /          \
!                                     |            \
!                                     |             \
!                                     |              |
!                   iy1_rule1  --------> x           |
!
!--------------------------------------------------------------------------
        integer*4 ith_avgpix                ! Thretholding level for determination
                                            ! of cardiac edges.                 [O]
                                            ! This threshold level is determined
                                            ! by histogram analysis of the center
                                            ! quadrant area.
        integer*2 image1(ncol,nlin)         ! Original Image1(current image)    [I]
        integer*4 giveangle                 ! SW for shift vector selection,    [I]
                                            !  1 => Dominant shift vector
                                            ! -1 => Non dominant shift vector
        integer*2 midline                   ! Midline of image1                 [I]
        integer*4 iflag                     ! Vector flip switch                [I]
                                            ! iflag =  0 -> Use dominant orientation
                                            ! for fitting.
                                            ! iflag = -1 -> Use not dominant orientation
                                            ! for fitting.
                                            ! iflag =  1 -> Switch vector for R lung
                                            ! iflag =  2 -> Switch vector for L lung
        integer*4 iterative                 ! Loop counter of warping iteration [I]
        integer*2 toplung                   ! Y-location of Lung top            [I]
        integer*4 MAXPT                     ! Maximum No. of ROI Pairs          [I]
!--------------------------------------------------------------------------
!       VARIABLES
!--------------------------------------------------------------------------
        character str*80
        integer*4 npx, npy                  ! MATRIX SIZE OF SHIFT-MAPS
        integer*4 IDX(MAXPT), IDY(MAXPT)! Initial Shift Values by Cross-correlation
                                            ! formatted for the rectanglar shift-value
map.
        integer*4 IDX2(MAXPT), IDY2(MAXPT)! Initial Shift Values by
Cross-correlation
```

```
                                            ! formatted for the rectanglar shift-value
map.
                                            ! These are copy of IDX,IDY.
        integer*4 tpco(2,MAXPT), saco(2,MAXPT)! Center location for Template ROI
                                            ! and Search area ROI
                                            ! formatted for the rectangler
center-location maps.
        integer*4 pstart(2), dis(2)
        integer*4 avex, avey                ! Average for shift vector DX and DY
        integer*4 C, L, I, J, N, m
        real*4    im1
        real*4    delx, dely, pi
        real*4    ll,kk
        real*4    weight(MAXPT)             ! Weights for surface fitting
                                            ! formatted for the rectangler weight maps.
        real*4    CCo(MAXPT)                ! Cross-Correlation Values
                                            ! formatted for the rectangler CC map.
        real*4    angle(MAXPT)              ! Shift vector orientations
                                            ! formatted for the rectangler orientation
map.
        real*4    anglewk(MAXPT)            ! Shift vector orientations arrenged to
                                            ! select dominant vectors easily.
        real*4    roiavg(MAXPT)             ! Avg pixel value of ROIs.
        real*4    roisd(MAXPT)              ! Avg SD of pixel value of ROIs.
        real*4    avg,sd                    ! Avg and SD of roisd(MAXPT).
        real*4    weightlow                 ! weight value for mediastinum area
        integer*4 histo(MAXPT)              ! Histogram of roisd(MAXPT).
        integer*4 iroisd,inum
        integer*4 ignorenum                 ! Number of excluding pixels for
                                            ! determination of Avg pixel value of ROIs.
        integer*4 id, leng
        integer*4 i_X,i_Y
        integer*4 iori
        real*4    ohist_r(0:359),ohist_l(0:359) ! Accumlated shift vector
orientation
                                            ! histogram for each lung.
        real*4    vec
        real*4    imaxhist_r,imaxhist_l ! Peak values of the shift vector
orientation
                                            ! histogram.
        integer*4 imaxhist_r_x,imaxhist_l_x  ! Peak angles of the shift vector
                                            ! orientation histogram.
        integer*4  k
        real*4     avghist_r,avghist_l      ! Avg of shift vector orientation histogram.
                                            ! This is Avg of frequency.
        common    /LOGFILE/ id
!--------------------------------------------------------------------------
!       FUNCTIONS
!--------------------------------------------------------------------------
c       integer*4 Create_Weighting_Factor_Map
c       integer*4 PutOutputF, UTL$STR_PRINT, UTL$FILE_WRITE
!--------------------------------------------------------------------------
!       PARAMETERS
!--------------------------------------------------------------------------
        pi=3.14159265
        weightlow=0.25
!
!--------------------------------------------------------------------------
!       1. DETERMINE THE MATRIX SIZE OF SHIFT-MAPS
!--------------------------------------------------------------------------
        npx = ( region2(3) - region2(1) ) / inc + 1
        npy = ( region2(4) - region2(2) ) / inc + 1
        pstart(1) = region2(1)
```

```
       pstart(2) = region2(2)
       dis(1) = inc
       dis(2) = inc
!-----------------------------------------------------------------------------
!      2. Initialize of dimensions
!-----------------------------------------------------------------------------
       do l=1,MAXPT
           IDX(l)=0
           IDY(l)=0
           IDX2(l)=0
           IDY2(l)=0
           weight(l)=0.0
           CCo(l)=0.0
           angle(l)=0.0
           anglewk(l)=0.0
           roiavg(l)=0.0
           roisd(l)=0.0
           histo(l)=0.0
       end do
       do l=0,359
           ohist_r(l)=0.0
           ohist_l(l)=0.0
       end do
!-----------------------------------------------------------------------------
!      3. CREATE SHIFT-MAPS
!-----------------------------------------------------------------------------
       N = 1
       avex = 0
       avey = 0
       do L = 1, npy
           do C = 1, npx
               I = region2(1) + inc * ( C - 1 )
               J = region2(2) + inc * ( L - 1 )
!-----------------------------------------------------------------------------
!              3-1. POINTS USED AS CENTERS OF TEMPLATES
!-----------------------------------------------------------------------------
               if ( N.le.number .and.
       1            tpc(1,N).eq.I .and. tpc(2,N).eq.J ) then
                   IDX(C+(L-1)*npx) = DX(N)
                   IDY(C+(L-1)*npx) = DY(N)
                   tpco(1,C+(L-1)*npx) = tpc(1,N)
                   tpco(2,C+(L-1)*npx) = tpc(2,N)
                   saco(1,C+(L-1)*npx) = sac(1,N)
                   saco(2,C+(L-1)*npx) = sac(2,N)
                   CCo(C+(L-1)*npx) = CC(N)
                   if(CCo(C+(L-1)*npx).lt.0) CCo(C+(L-1)*npx)=0.0
                   avex = avex + DX(N)
                   avey = avey + DY(N)
                   N = N + 1
!-----------------------------------------------------------------------------
!              3-2. POINTS NOT USED AS CENTERS OF TEMPLATES
!-----------------------------------------------------------------------------
               else
                   IDX(C+(L-1)*npx) = 0
                   IDY(C+(L-1)*npx) = 0
                   tpco(1,C+(L-1)*npx) = 0
                   tpco(2,C+(L-1)*npx) = 0
                   saco(1,C+(L-1)*npx) = 0
                   saco(2,C+(L-1)*npx) = 0
                   CCo(C+(L-1)*npx) = 0.0
               end if
           end do
       end do
```

```
       avex = avex / number
       avey = avey / number
!-----------------------------------------------------------------------------
!    Preparation for segmentation of lung area.
!    Determine average pixel value and SD of the pixel value for ROIs of image1
!-----------------------------------------------------------------------------
       do i=1,MAXPT
                roiavg(i)=0.0
                roisd(i)=0.0
                weight(i)=1.0
       end do
       do i=1,MAXPT
          if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
                   ignorenum=0
              do m=saco(2,i)-16,saco(2,i)+16
                 do l=saco(1,i)-16,saco(1,i)+16
                    if(m.lt.1.or.m.gt.nlin)then
                         ignorenum=ignorenum+1
                    else
                         im1=float(image1(l,m))
                         roiavg(i)=roiavg(i)+im1
                    end if
                 end do
              end do
              roiavg(i)=roiavg(i)/(33.0*33.0-ignorenum)
          end if
       end do

       do i=1,MAXPT
          if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
                   ignorenum=0
              do m=saco(2,i)-16,saco(2,i)+16
                 do l=saco(1,i)-16,saco(1,i)+16
                    if(m.lt.1.or.m.gt.nlin)then
                         ignorenum=ignorenum+1
                    else
                     im1=float(image1(l,m))-roiavg(i)
                     roisd(i)=roisd(i)+im1**2
                    end if
                 end do
              end do
       roisd(i)=sqrt(roisd(i)/((33.0*33.0-ignorenum)-1))
          else
             weight(i)=0.0
          end if
       end do
!-----------------------------------------------------------------------------
!      Determination of histogram of SD for ROIs
!-----------------------------------------------------------------------------
       inum=0
       do i=1,MAXPT
           if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
!-----------------------------------------------------------------------------
!              area #1
!-----------------------------------------------------------------------------
               if(saco(2,i).gt.iy1_rule1) then
                   inum=inum+1
                   iroisd=int(roisd(i))
                   histo(iroisd)=histo(iroisd)+1
               end if
!-----------------------------------------------------------------------------
!              area #2
!-----------------------------------------------------------------------------
```

```
               if(saco(1,i).ge.ix1_rule2.and.saco(1,i).lt.ix2_rule2.
     &             and.saco(2,i).lt.iy2_rule2) then
                   inum=inum+1
                   iroisd=int(roisd(i))
                   histo(iroisd)=histo(iroisd)+1
               end if
!-----------------------------------------------------------------------------
!              area #3
!-----------------------------------------------------------------------------
               if(saco(2,i).ge.iy2_rule2.and.saco(2,i).lt.iy1_rule1.
     &             and.saco(1,i).ge.ix1_rule3(1,saco(2,i)).and.
     &             saco(1,i).lt.ix1_rule3(2,saco(2,i))) then
                   inum=inum+1
                   iroisd=int(roisd(i))
                   histo(iroisd)=histo(iroisd)+1
               end if

           end if
       end do

!-----------------------------------------------------------------------------
!      Determination of average and SD of histogram of SD within ROIs.
!-----------------------------------------------------------------------------
       avg=0.0
       do l=1,MAXPT
          avg=avg+float(histo(l)*l)
       end do
          avg=avg/float(inum)
          sd=0.0
       do l=1,MAXPT
          sd=sd+((float(l)-avg)**2)*float(histo(l))
       end do
          sd=sqrt(sd/float(inum-1))
c       write(*,*) avg,sd,'www'
!-----------------------------------------------------------------------------
!      Determination ROIs within lung area
!      If ROI is located in lungs, weight(i)=1.0
!                         othwise, weight(i)=0.0
!-----------------------------------------------------------------------------
       do i=1,MAXPT
           if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
!-----------------------------------------------------------------------------
!              area #1
!-----------------------------------------------------------------------------
               if(saco(2,i).gt.iy1_rule1) then
                if(roiavg(i).gt.ith_avgpix.or.roisd(i).lt.avg+sd)then
                   weight(i)=weightlow
                else
                   weight(i)=1.0
                end if
               end if
!-----------------------------------------------------------------------------
!              area #2
!-----------------------------------------------------------------------------
               if(saco(1,i).ge.ix1_rule2.and.saco(1,i).lt.ix2_rule2.
     &             and.saco(2,i).lt.iy2_rule2) then
                if(roiavg(i).gt.ith_avgpix.or.roisd(i).lt.avg+sd)then
                      weight(i)=weightlow
                   else
                      weight(i)=1.0
                   end if
               end if
!-----------------------------------------------------------------------------
```

```
!              area #3
!-----------------------------------------------------------------------------
               if(saco(2,i).ge.iy2_rule2.and.saco(2,i).lt.iy1_rule1.
     &             and.saco(1,i).ge.ix1_rule3(1,saco(2,i)).and.
     &             saco(1,i).lt.ix1_rule3(2,saco(2,i))) then
               if(roiavg(i).gt.ith_avgpix.or.roisd(i).lt.avg+sd)then
                      weight(i)=weightlow
                   else
                      weight(i)=1.0
                   end if
               end if
!-----------------------------------------------------------------------------
!              area #4
!-----------------------------------------------------------------------------
               if(saco(2,i).gt.iy1_rule1+50) then
                      weight(i)=weightlow
               end if

           end if
       end do
c       do C=1,npy
c          write(*,998) (weight(L+(C-1)*npx),l=1,npx)
c       end do
!-----------------------------------------------------------------------------
!              Determine angle of shift vectors for each ROI
!-----------------------------------------------------------------------------
       do L = 1, npy
           do C = 1, npx
                  i_X=IDX(C+(L-1)*npx)-shiftmid
                  i_Y=IDY(C+(L-1)*npx)-ribtop
                  delx=float(i_X)
                  dely=float(i_Y)
                  angle(C+(L-1)*npx)=atan2(-dely,-delx)/pi*180.0
                  if(angle(C+(L-1)*npx).lt.0.)
     &         angle(C+(L-1)*npx)=360.0 + angle(C+(L-1)*npx)
                  if(angle(C+(L-1)*npx).ge.360.)
     &         angle(C+(L-1)*npx)= angle(C+(L-1)*npx) - 360.0
           end do
       end do

       if(iterative.ge.2) goto 5555
!-----------------------------------------------------------------------------
!      Make shift vector weighted vector orientation histogram
!-----------------------------------------------------------------------------
               do i=1,MAXPT
                    if(weight(i).eq.1.0) then
                    if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
                         if(saco(1,i).lt.midline) then
                            iori=int(angle(i))
               vec=sqrt(float((IDX(i)-shiftmid)**2+(IDY(i)-ribtop)**2))
                            ohist_r(iori)=ohist_r(iori)+vec
                         else
                            iori=int(angle(i))
               vec=sqrt(float((IDX(i)-shiftmid)**2+(IDY(i)-ribtop)**2))
                            ohist_l(iori)=ohist_l(iori)+vec
                         end if
                    end if
                    end if
               end do
!-----------------------------------------------------------------------------
!      Determination of the peak of shift vector orientation histogram
!-----------------------------------------------------------------------------
               avghist_r=0.0
```

```
            avghist_l=0.0
            imaxhist_r=-1000.0
            imaxhist_l=-1000.0
            do i=0,330,30
                  ll=0.0
                  do j=0,29
                  ll=ll+ohist_r(i+j)
                  end do
             if(ll.gt.imaxhist_r) then
                    imaxhist_r=ll
                    imaxhist_r_x=i
             end if
                  ll=0.0
                  do j=0,29
                  ll=ll+ohist_l(i+j)
                  end do
             if(ll.gt.imaxhist_l) then
                    imaxhist_l=ll
                    imaxhist_l_x=i
             end if
                  ll=0.0
                  kk=0.0
                  do j=0,29
                  ll=ll+ohist_r(i+j)
                  kk=kk+ohist_l(i+j)
                  end do
                avghist_r=avghist_r+ll
                avghist_l=avghist_l+kk
            end do
                avghist_r=avghist_r/12.0
                avghist_l=avghist_l/12.0

c       open(1,file='hist_R')
c               do i=0,330,30
c                     ll=0.0
c                     do j=0,29
c                     ll=ll+ohist_r(i+j)
c                     end do
c                     write(1,*) i,int(ll)
c                end do
c       close(1)
c       open(2,file='hist_L')
c               do i=0,330,30
c                     kk=0.0
c                     do j=0,29
c                     kk=kk+ohist_l(i+j)
c                     end do
c                     write(2,*) i,int(kk)
c               end do
c       close(2)

!-----------------------------------------------------------------------
!       If peak level is less than avghist_r*2,   0-180 => Dominant,
!                                               180-360 => Non dominant.

          if(avghist_r*2.gt.imaxhist_r) imaxhist_r_x=90
          if(avghist_l*2.gt.imaxhist_l) imaxhist_l_x=90
!-----------------------------------------------------------------------

!-----------------------------------------------------------------------
!       Determination weights based on shift vector analysis
!-----------------------------------------------------------------------
          do i=1,MAXPT
```

```
          if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
              if(weight(i).eq.1.0) then
              if(saco(1,i).lt.midline) then
               anglewk(i)=angle(i)-(imaxhist_r_x-90)
          if(anglewk(i).lt.0) anglewk(i)=anglewk(i)+360
          if(anglewk(i).ge.360) anglewk(i)=anglewk(i)-360
              else
               anglewk(i)=angle(i)-(imaxhist_l_x-90)
          if(anglewk(i).lt.0) anglewk(i)=anglewk(i)+360
          if(anglewk(i).ge.360) anglewk(i)=anglewk(i)-360
              end if
              end if
          end if
  end do
  if(giveangle.eq.1.and.iflag.eq.0) then
         do i=1,MAXPT
          if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
             if(weight(i).eq.1.0) then
                  if(anglewk(i).ge.180) weight(i)=0.0
             end if
          end if
         end do
   end if
  if(giveangle.eq.-1.and.iflag.eq.0) then
         do i=1,MAXPT
          if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
              if(weight(i).eq.1.0) then
                 if(anglewk(i).lt.180) weight(i)=0.0
              end if
          end if
         end do
   end if
  if(giveangle.eq.1.and.iflag.eq.1) then
         do i=1,MAXPT
          if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
              if(weight(i).eq.1.0) then
             if(saco(1,i).lt.midline) then
                    if(anglewk(i).lt.180) weight(i)=0.0
                else
                    if(anglewk(i).ge.180) weight(i)=0.0
                end if
             end if
             end if
         end do
   end if
  if(giveangle.eq.-1.and.iflag.eq.1) then
         do i=1,MAXPT
          if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
              if(weight(i).eq.1.0) then
             if(saco(1,i).lt.midline) then
                    if(anglewk(i).ge.180) weight(i)=0.0
                else
                    if(anglewk(i).lt.180) weight(i)=0.0
                end if
             end if
             end if
         end do
   end if
  if(giveangle.eq.1.and.iflag.eq.2) then
         do i=1,MAXPT
          if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
              if(weight(i).eq.1.0) then
             if(saco(1,i).lt.midline) then
```

```
                        if(anglewk(i).ge.180) weight(i)=0.0
                    else
                        if(anglewk(i).lt.180) weight(i)=0.0
                    end if
                 end if
                 end if
              end do
          end if
         if(giveangle.eq.-1.and.iflag.eq.2) then
              do i=1,MAXPT
               if(tpco(1,i).ne.0.and.tpco(2,i).ne.0) then
                   if(weight(i).eq.1.0) then
                  if(saco(1,i).lt.midline) then
                        if(anglewk(i).lt.180) weight(i)=0.0
                    else
                        if(anglewk(i).ge.180) weight(i)=0.0
                    end if
                 end if
                 end if
              end do
          end if
 5555   continue

!-----------------------------------------------------------------------
!       5. FITTING AND INTERPOLATIONS
!-----------------------------------------------------------------------
        if(iterative.eq.1) then
            call Weighted_Fit_Prepare( npx, npy, order, weight )
        end if

          do i=1,MAXPT
                IDX2(i)=IDX(i)
                IDY2(i)=IDY(i)
          end do
        if(iterative.eq.1) then
c       write(*,*) '...Fitting Shift Value DX.'
        call Weighted_Fit_Intp( IDX2, FX, npx, npy, ncol, nlin, pstart, dis,
        1                                order, avex, weight,FITX)

c       write(*,*) '...Fitting Shift Value DY.'
        call Weighted_Fit_Intp( IDY2, FY, npx, npy, ncol, nlin, pstart, dis,
        1                                order, avey, weight,FITY)

        else
c       write(*,*) '...Fitting Shift Value DX.'
        call Weighted_nofit_Intp( IDX2, FX, npx, npy, ncol, nlin, pstart, dis,
        1        order, avex, weight,FITX)

c       write(*,*) '...Fitting Shift Value DY.'
        call Weighted_nofit_Intp( IDY2, FY, npx, npy, ncol, nlin, pstart, dis,
        1       order, avey, weight,FITY)
        end if

!-----------------------------------------------------------------------
!       6. WRITE PARAMETERS
!-----------------------------------------------------------------------
c       leng = UTL$STR_PRINT( str, ' Fitting by %2ith Order', order )
c       call UTL$FILE_WRITE( id, str )
c       leng = UTL$STR_PRINT
c       1   ( str, ' (%4.2f,%4.2f)-(%4.2f,%4.2f)-(%4.2f,%4.2f)-(%4.2f,%4.2f)',
c       2     WF(1,1),WF(2,1),WF(1,2),WF(2,2),WF(1,3),WF(2,3),WF(1,4),WF(2,4) )
c       call UTL$FILE_WRITE( id, str )
        write(id,*) ' Fitting by ', order, 'th Order'
```

```
        write(id, FMT = '(F4.2,F4.2)-(F4.2,F4.2)-(F4.2,F4.2)-F(4.2,F4.2)')
     1                          WF(1,1), WF(2,1), WF(1,2), WF(2,2),
     2                          WF(1,3), WF(2,3), WF(1,4), WF(2,4)
!-----------------------------------------------------------------------
!       END
!-----------------------------------------------------------------------
        return
        end
!
!=======================================================================
!***********************************************************************
!                  This subroutine is not used in this version.
!***********************************************************************
        subroutine Create_Weighting_Factor_Map( number, tpc, CC, WF, npx, npy,
        1                                 inc, region2, weight )
!=======================================================================
!       Ver. 1.0
!       Written by Akiko Kano,  Dec.29, 1992
!-----------------------------------------------------------------------
!       This function creates a weighting factor map with the size of npx x npy.
!
!       (1) Weighting factors for fitting are determined based on CC, using the
!           LUT given by "WF".
!       (2) Points which were not used as centers of templates are given with
!           weighting factor "0".
!=======================================================================
!       ARGUMENTS
!-----------------------------------------------------------------------
        implicit  none
c       integer*4 Create_Weighting_Factor_Map
        integer*4 number                  ! Number of Templates            [I]
        integer*4 tpc(2,*)                ! Centers of Templates on Image2  [I]
        real*4    CC(*)                   ! Cross-Correlation Values        [I]
        real*4    WF(2,11)                ! LUT of Weighting Factor vs. Cross-
                                          !  Correlation Value              [I]
        integer*4 npx, npy                ! Matrix Size of Shift-Map        [I]
        integer*4 inc                     ! Distance Increment of ROIs(Pixels)[I]
        integer*4 region2(4)              ! Smallest Rect. Area Including tpc [I]
        real*4    weight(npx,npy)         ! Weighting Factor                [O]
        integer*4 LEVEL,iwk
!-----------------------------------------------------------------------
!       VARIABLES
!-----------------------------------------------------------------------
        parameter (LEVEL=1000)
        integer*4 C, L, I, J, N
        real*4    IWF(0:LEVEL)
!-----------------------------------------------------------------------
!       BEGIN
!-----------------------------------------------------------------------
        call Interpolate_LUT_Adpt( LEVEL, WF, IWF ,CC, npx, npy)
        N = 1
        do L = 1, npy
            do C = 1, npx
                I = region2(1) + inc * ( C - 1 )
                J = region2(2) + inc * ( L - 1 )
                if ( N.le.number .and.
     1              tpc(1,N).eq.I .and. tpc(2,N).eq.J ) then
                    weight(C,L) = IWF( int(CC(N)*float(LEVEL)) )
                    N = N + 1
                else
                    weight(C,L) = 0.0
                end if
            end do
        end do
!-------------------------------------------------------------------------------
!       END
!-------------------------------------------------------------------------------
        return
        end
!
!===============================================================================
!*******************************************************************************
!                    This subroutine is not used in this version.
!*******************************************************************************
        subroutine Interpolate_LUT_Adpt( level, WF, IWF, CC, nx, ny )
!===============================================================================
!       Ver. 1.0
!       Written by Akiko Kano,  Dec.29, 1992
!-------------------------------------------------------------------------------
!       This subroutine interpolates a LUT for weighting factors.
!===============================================================================
!       ARGUMENTS
!-------------------------------------------------------------------------------
        implicit  none
        integer*4 level                   ! No. of Levels                     [I]
        real*4    WF(2,11)                ! LUT of Weighting Factor vs. Cross-
                                          !  Correlation Value                [I]
        real*4    IWF(0:level)            ! Interpolated LUT                  [O]
        real*4    CC(*)                   ! Cross-Correlation Value           [I]
        integer*4 nx                      ! Columns of ROIs                   [I]
        integer*4 ny                      ! Rows of ROIs                      [I]
!-------------------------------------------------------------------------------
!       VARIABLES
!-------------------------------------------------------------------------------
        integer*4 IC(11)
        integer*4 N, I, J
        real*4    MAX_CC, MIN_CC
        integer*4 H_CC(100)
        integer*4 count
        integer*4 H_level
        integer*4 H_sum, SUM
        integer*4 P1, P2, H1, H2
        parameter (P1 = 60, P2 = 80)
!-------------------------------------------------------------------------------
!       BEGIN
!-------------------------------------------------------------------------------
        MAX_CC = 0.0
        MIN_CC = 1.0

        do I = 1, 100
                H_CC(I) = 0
        end do
        count = 1
        do I = 1, ny
            do J = 1, ny
                H_level = 100*CC(count)
c               write(*,*) 'H_level = ', H_level
                if(H_level.lt.1) H_level=1
                H_CC(H_level) = H_CC(H_level) + 1
                count = count + 1
            end do
        end do

        SUM = 0
        do I = 1, 100
                SUM = SUM + H_CC(I)
```

```
c               write(*,*) 'H_CC(',I,') = ', H_CC(I), '    SUM = ', SUM
        end do

        H_sum = 0
        I = 1

        H1 = int( float(P1)*SUM/100)
        do while ( H_sum .lt. H1 )
            H_sum = H_sum + H_CC(I)
            I = I + 1
        end do
        WF(1,2) = (I-1)*0.01
c       write(*,*) P1, '% = ', WF(1,2)

        H2 = int( float(P2)*SUM/100 )
        do while ( H_sum .lt. H2 )
            H_sum = H_sum + H_CC(I)
            I = I + 1
        end do
        WF(1,3) = (I-1)*0.01
c       write(*,*) P2, '% = ', WF(1,3)

c       write(*,*) 'H1 = ', H1, '    H2 = ', H2

        IC(1) = 0
        N = 1
        do while ( WF(1,N).lt.1.0 )
            N = N + 1
            IC(N) = int( float(level) * WF(1,N) )
        end do

        IWF(0) = WF(2,1)
        do I = 1, N-1
            do J = IC(I)+1, IC(I+1)
                IWF(J) = WF(2,I)
        1                     + (WF(2,I+1)-WF(2,I)) * (J-IC(I)) / (IC(I+1)-IC(I))
            end do
        end do
!-------------------------------------------------------------------------------
!       END
!-------------------------------------------------------------------------------
        return
        end
```

```
c=======================================================================

        subroutine smooth (x,n,s)
c
c       smooth by using moving average
c       input:  s       range for average(odd number)
c               n       No. of elements
c               x(n)    Input data & output data
c
c       1/24/92         Hong Jia
c                       Based on Shige's fax program 12/20/91
c
c=======================================================================

        integer*2 n,s,w,i,j,k
        integer*2 b1,b2
        real*4  x(n),y(512),sum

c       ---------------------
c       Smoothing
c       ---------------------

        w=s/2

        b1=w+1
        b2=n-w

        do i=b1,b2
           sum=0.0
           do j=1,s
           sum=sum+x(i-w+j-1)
           end do
           x(i)=sum/float(s)
        end do

c       do i=b1,b2
c       x(i)=y(i)
c       end do

        return
        end
```

```
c      -----------------------------------------------------
c      Sobel filter
c      -----------------------------------------------------
        subroutine sobel2(ip,sb,id,ix,iy,iv)

!       ip(ix,iy)       : input image                    [I]
!       sb(ix,iy)       : sobel output                   [O]
!       id(ix,iy)       : orientation output             [O]
!       ix,iy           : matrix size of x and y         [I]
!       iv              : cwinterval for sobel operation [I]

        implicit  integer*4 (i-n)
        integer*2 ip(ix,iy),sb(ix,iy)
        real*4    id(ix,iy)

        pi=3.14159265

c      -----------------------------------------------------
c      Sobel filtering
c      -----------------------------------------------------
        do 50 j=iv+1,iy-iv
          do 40 i=iv+1,ix-iv

          dx1=2*ip(i-iv,j)+ip(i-iv,j-iv)+ip(i-iv,j+iv)
          dx2=2*ip(i+iv,j)+ip(i+iv,j-iv)+ip(i+iv,j+iv)
          dx=dx2-dx1
          dy1=2*ip(i,j+iv)+ip(i+iv,j+iv)+ip(i-iv,j+iv)
          dy2=2*ip(i,j-iv)+ip(i+iv,j-iv)+ip(i-iv,j-iv)
          dy=dy2-dy1

          sb(i,j)=nint(sqrt(dx*dx+dy*dy))

          id(i,j)=atan2(dy,dx)/pi*180.0
          if(id(i,j).lt.0.) id(i,j)=360.0+id(i,j)

   40   continue
   50    continue


        return
        end
```

```
/*===================================================================
      Utilities for Temporal Subtraction
      Coded by Shige
Following functions are included,
void                 getDirectory
int                  trimcha
char                 *getExtension
void                 cleanString
int                  wsplit
StudyPara            ReadTSubDefFile
void                 CheckDefFile
NumTime              GetNumTime
TexTime              GetTexTime
int                  getFileSize
InputImageFormat     getImageFormat
GenStand             getGenStandInfo
void                 getSFCRImage       //short image
void                 getSDCMImage       //short image
void                 getSTHVImage       //short image
void                 getSNOHImage       //short image
void                 getSJDBImage       //short image
void                 getSGENImage       //short image
void                 getCFCRImage       //unsigned char image
void                 getCDCMImage       //unsigned char image
void                 getCTHVImage       //unsigned char image
void                 getCNOHImage       //unsigned char image
void                 getCJDBImage       //unsigned char image
void                 getCGENImage       //unsigned char image
void                 getODCMImage       //Original image reduced
====================================================================*/
#include     <stdio.h>
#include     <string.h>
#include     <stdlib.h>
#include     <ctype.h>
#include     <time.h>
#include     <math.h>
#include     "TempSub.H"

/*==============================================================
Name      : getDirectory
Function : Get Directory
Date      : 04/30/98
===============================================================*/
void  getDirectory(char *dir, char *fname)
{
  char            c;
  unsigned int    n;
  char            *p1, *p2;

  c  = '/';
  p1 = fname;
  p2 = strrchr(fname, c);

  if (p2 == NULL) strcpy(dir, "./");
  else {
    n = (int)(p2 - p1 + 1);
    strncpy(dir, fname, n);
    dir[n] = 0;
    }
}
```

```
/*=======================================================================
==
Name      : trimcha
Function : Trim string between two characters(last c1 and first c2)
         Input     s_source: String to be trimmed
                   c1, c2   : two characters
         Output    s_dest   : Output String
Usage     : int trimcha(s_dest, s_source, c1, c2)
Return    : Number of Characters for Output String
Author    : Shige
Date      : 8/4/96
=======================================================================
==*/
int   trimcha(char *s_dest, char *s_source, char c1, char c2)
{
  char       r0[128], r1[128], r2[128];
  int nchar;

  nchar = 0;

  if (strrchr(s_source, c1) == NULL) strcpy(r1, s_source);
  else                              strcpy(r1, strrchr(s_source, c1) +1);

  if (strchr(s_source, c2) == NULL) {
    nchar = strlen(r1);
    }
  else {
    strcpy(r2, strchr(s_source, c2));
    nchar = strlen(r1) - strlen(r2);
    }

  if (nchar < 1) {
    nchar = 0;
    return(nchar);
    }

  strcpy(s_dest, r1);
  *(s_dest + nchar) = NULL;
  return(nchar);
}

/*=======================================================================
==
Name      : getExtension
Function : Get Extension
Input     : fname
Output         : ext and return value(pointer)

Usage Example   : char Extension[60];

                [Extension = ]getExtension(Extension, FName);
                or
                printf("%s", getExtension(Extension, FName);

Date      : 04/30/98
=======================================================================
==*/
char   *getExtension(char *ext, char *fname)
{
char         c;
int          i, n, nc, m;
char         *p1, *p2;

c  = '.';
nc = strlen(fname);
p1 = fname;
p2 = strrchr(fname, c);

if (p2 == NULL) ext[0] = 0;
else {
  n = (int)(p2 - p1 + 1);
  m = nc - n + 1;
  for (i = 0; i < m; i++) ext[i] = fname[n - 1 + i];
  ext[m] = 0;
  }
return ext;
}

/*=======================================================================
==
Name      : cleanString
Function : (1) Remove characters except for printable
          (2) Remove heading spaces and tabs
          (3) Remove ending spaces and tabs
[I]/[O]        : *PatName

Date      : 11/05/98
=======================================================================
==*/
void  cleanString(char *PatName)
{
int   i, c, n;

n = strlen(PatName);
c = 0;
for (i = 0; i < n; i++) {
  if ((isprint(PatName[i]) != 0 && PatName[i] > 0) ||
       isspace(PatName[i]) != 0)
       PatName[c++] = PatName[i];
  }
PatName[c] = 0;

n = c;
for (i = 0; i < n; i++) {
  if (isspace(PatName[i]) != 0) PatName[i] = 0;
  else                                  break;
  }
for (i = n - 1; i >= 0; i--) {
  if (isspace(PatName[i]) != 0) PatName[i] = 0;
  else                              break;
  }

c = 0;
for (i = 0; i < n; i++) {
  if (PatName[i] != 0) PatName[c++] = PatName[i];
  }
PatName[c] = 0;
}

/*===================================================================
To split words from a character buffer
Return (int)Number of Words
CR(\n) character is removed.
Address of each word is put in arg_v[]

int arg_c = wsplit(str, str_buf, arg_v)
arg_c         Number of words
str           Buffer containing many words
str_buf             Work Buffer separated with NULL
arg_v[]             Addresss of each word

4/19/95       coded by SHIGE
6/07/95       modified by SHIGE
====================================================================*/
#define           ARG_IN             0
#define           ARG_OUT            1
#define           MAXSIZE            1024

int   wsplit (char *str, char *str_buf, char *arg_v[])
{
  int i, j, flag;
  char      c;
  char      *str_p;

  str_p = strcpy(str_buf, str);

  for ( i = j = 0, flag = ARG_OUT ; i < MAXSIZE ; i++ ) {
    c = *(str_p + i);
    if ( c == '\0' || c == '\n') {
      *(str_p + i) = NULL;
      arg_v[j]     = NULL;
      break;
      }

    if ( c == ' ' || c == '\t' ) {
      *(str_p + i) = NULL;
      flag = ARG_OUT;
      continue;
      }

    if ( flag == ARG_OUT ) {
      arg_v[j++] = str_p + i;
      flag = ARG_IN;
      continue;
      }
    }
  return (j);
}

/*==============================================================
Read Definition File of Temporal Subtraction
INPUT DEf. File Name
OUTPUT      StudyPara defined in workstation.h
11/16/98 Code by Shige
===============================================================*/
StudyPara ReadTSubDefFile(char *DefFile)
{
FILE         *fp;
StudyPara    studyPara;

int   wsplit(char*, char*, char**);
char  str[MAXSIZE], str_buf[MAXSIZE];
int   nword;
```

```
char  *str_p[100];

/*----------------------------------------------------
      Open DefFile
------------------------------------------------------*/
if ((fp = fopen(DefFile, "r")) == 0) {
    printf("\7\7%s is missing \n",DefFile); exit(0);
  }

/*----------------------------------------------------
      Read line by line
------------------------------------------------------*/
while ( (fgets(str, MAXSIZE, fp)) != 0) {
  if (str[0] == '\\')   break;
  if (str[0] == '#' )   continue;
  if (str[0] == '\n')   continue;

  /*--------------------------------------------------
            Word Split
  ----------------------------------------------------*/
  nword = wsplit(str, str_buf, str_p);
  if (nword < 2) continue;

  /*----------------------------------------------------
      InputImageFormat
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "InputImageFormat")) == 0) {
    if          ((strcmp(str_p[1],"INPUT_STANDARD_FCR_HEADER")) == 0)
            studyPara.inputImageFormat = INPUT_STANDARD_FCR_HEADER;
    else if ((strcmp(str_p[1],"INPUT_DICOM_FCR_HEADER")) == 0)
            studyPara.inputImageFormat = INPUT_DICOM_FCR_HEADER;
    else if ((strcmp(str_p[1],"INPUT_DICOM_THV_HEADER")) == 0)
            studyPara.inputImageFormat = INPUT_DICOM_THV_HEADER;
    else if ((strcmp(str_p[1],"INPUT_NON_HEADER")) == 0)
            studyPara.inputImageFormat = INPUT_NON_HEADER;
    else {
      printf("Wrong InputImageFormat in %s\n", DefFile); exit(0);
      }
    }

  /*----------------------------------------------------
      Matrix Size
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "NoOfColumn")) == 0)
                 studyPara.col = atoi(str_p[1]);
  if ((strcmp(str_p[0], "NoOfLine")) == 0)
                 studyPara.lin = atoi(str_p[1]);

  /*----------------------------------------------------
      OutputImageFormat
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "OutputImageFormat")) == 0) {
    if          ((strcmp(str_p[1],"OUTPUT_STANDARD_FCR_HEADER")) == 0)
            studyPara.outputImageFormat = OUTPUT_STANDARD_FCR_HEADER;
    else if ((strcmp(str_p[1],"OUTPUT_DICOM_THV_HEADER")) == 0)
            studyPara.outputImageFormat = OUTPUT_DICOM_THV_HEADER;
    else if ((strcmp(str_p[1],"OUTPUT_NON_HEADER")) == 0)
            studyPara.outputImageFormat = OUTPUT_NON_HEADER;
    else {
      printf("Wrong OutputImageFormat in %s\n", DefFile); exit(0);
      }
```

```
    }

  /*----------------------------------------------------
      SubFName
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "SubFName")) == 0) {
    if          ((strcmp(str_p[1],"SUB_FNAME_FIX")) == 0)
            studyPara.subFName = SUB_FNAME_FIX;
    else if ((strcmp(str_p[1],"SUB_FNAME_PATIENT_ID")) == 0)
            studyPara.subFName = SUB_FNAME_PATIENT_ID;
    else if ((strcmp(str_p[1],"SUB_FNAME_PATIENT_NAME_LONG")) == 0)
            studyPara.subFName = SUB_FNAME_PATIENT_NAME_LONG;
    else if ((strcmp(str_p[1],"SUB_FNAME_PATIENT_NAME_SHORT")) == 0)
            studyPara.subFName = SUB_FNAME_PATIENT_NAME_SHORT;
    else if ((strcmp(str_p[1],"SUB_FNAME_PATIENT_ID_DATE")) == 0)
            studyPara.subFName = SUB_FNAME_PATIENT_ID_DATE;
    else if ((strcmp(str_p[1],"SUB_FNAME_COMMAND_LINE")) == 0)
            studyPara.subFName = SUB_FNAME_COMMAND_LINE;
    else {
      printf("Wrong SubFName in %s\n", DefFile); exit(0);
      }
    }

  /*----------------------------------------------------
      SubFixFName
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "SubFixFName")) == 0)
                 strcpy(studyPara.subFixFName, str_p[1]);

  /*----------------------------------------------------
      DensityCorrection
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "DensityCorrection")) == 0) {
    if          ((strcmp(str_p[1],"OFF")) == 0)
            studyPara.densityCorrection = OFF;
    else if ((strcmp(str_p[1],"ON")) == 0)
            studyPara.densityCorrection = ON;
    else {
      printf("Wrong DensityCorrection in %s\n", DefFile); exit(0);
      }
    }

  /*----------------------------------------------------
      DenCorLutDir
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "DenCorLutDir")) == 0)
                 strcpy(studyPara.denCorLutDir, str_p[1]);

  /*----------------------------------------------------
      PreDirectory
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "PreDirectory")) == 0)
                 strcpy(studyPara.preDirectory, str_p[1]);

  /*----------------------------------------------------
      CurDirectory
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "CurDirectory")) == 0)
                 strcpy(studyPara.curDirectory, str_p[1]);

  /*----------------------------------------------------
```

```
      SubDirectory
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "SubDirectory")) == 0)
                 strcpy(studyPara.subDirectory, str_p[1]);

  /*----------------------------------------------------
      Combination
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "Combination")) == 0) {
    if          ((strcmp(str_p[1],"SINGLE_COMBINATION")) == 0)
            studyPara.combination = SINGLE_COMBINATION;
    else if ((strcmp(str_p[1],"MULTI_COMBINATION")) == 0)
            studyPara.combination = MULTI_COMBINATION;
    else {
      printf("Wrong Combination in %s\n", DefFile); exit(0);
      }
    }

  /*----------------------------------------------------
      IniDirectory
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "IniDirectory")) == 0)
                 strcpy(studyPara.iniDirectory, str_p[1]);

  /*----------------------------------------------------
      Rescue Program
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "RescueProg")) == 0) {
    if          ((strcmp(str_p[1],"OFF")) == 0)
            studyPara.rescueProg = OFF;
    else if ((strcmp(str_p[1],"ON")) == 0)
            studyPara.rescueProg = ON;
    else {
      printf("Wrong RescueProg in %s\n", DefFile); exit(0);
      }
    }

  /*----------------------------------------------------
      SubImageTransfer
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "SubImageTransfer")) == 0) {
    if          ((strcmp(str_p[1],"OFF")) == 0)
            studyPara.subImageTransfer = OFF;
    else if ((strcmp(str_p[1],"ON")) == 0)
            studyPara.subImageTransfer = ON;
    else {
      printf("Wrong SubImageTransfer in %s\n", DefFile); exit(0);
      }
    }

  /*----------------------------------------------------
      OrgImageTransfer
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "OrgImageTransfer")) == 0) {
    if          ((strcmp(str_p[1],"OFF")) == 0)
            studyPara.orgImageTransfer = OFF;
    else if ((strcmp(str_p[1],"ON")) == 0)
            studyPara.orgImageTransfer = ON;
    else {
      printf("Wrong OrgImageTransfer in %s\n", DefFile); exit(0);
      }
```

```
    }

  /*----------------------------------------------------
      SubDispPara
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "SubDispPara")) == 0) {
    if          ((strcmp(str_p[1],"BLACK_OUT")) == 0)
            studyPara.subDispPara = BLACK_OUT;
    else if ((strcmp(str_p[1],"OPTIMAL_CHEST")) == 0)
            studyPara.subDispPara = OPTIMAL_CHEST;
    else {
      printf("Wrong SubDispPara in %s\n", DefFile); exit(0);
      }
    }

  /*----------------------------------------------------
      PreDispPara
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "PreDispPara")) == 0) {
    if          ((strcmp(str_p[1],"BLACK_OUT")) == 0)
            studyPara.preDispPara = BLACK_OUT;
    else if ((strcmp(str_p(1],"OPTIMAL_CHEST")) == 0)
            studyPara.preDispPara = OPTIMAL_CHEST;
    else {
      printf("Wrong PreDispPara in %s\n", DefFile); exit(0);
      }
    }

  /*----------------------------------------------------
      PrgDirectory
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "PrgDirectory")) == 0)
                 strcpy(studyPara.prgDirectory, str_p[1]);

  /*----------------------------------------------------
      IntermediateResult
  ------------------------------------------------------*/
  if ((strcmp(str_p[0], "IntermediateResult")) == 0) {
    if          ((strcmp(str_p[1],"OFF")) == 0)
            studyPara.intermediateResult = OFF;
    else if ((strcmp(str_p[1],"ON")) == 0)
            studyPara.intermediateResult = ON;
    else {
      printf("Wrong IntermediateResult in %s\n", DefFile); exit(0);
      }
    }

  }   // End of while
fclose(fp);
return studyPara;
}

/*==========================================================
Check DefFile
11/20/98 Code by Shige
============================================================*/
void  CheckDefFile(StudyPara sdy)
{
if (sdy.inputImageFormat == INPUT_STANDARD_FCR_HEADER) {
  printf("\a\aINPUT_STANDARD_FCR_HEADER is not permitted!!\n"); exit(0);
  }
```

```
else if (sdy.inputImageFormat == INPUT_DICOM_FCR_HEADER) {
  if (sdy.outputImageFormat != OUTPUT_NON_HEADER) {
      printf("Wrong InputImageFormat\n"); exit(0);
      }
  }
else if (sdy.inputImageFormat == INPUT_DICOM_THV_HEADER) {
  if (sdy.outputImageFormat == OUTPUT_STANDARD_FCR_HEADER ) {
      printf("Wrong InputImageFormat\n"); exit(0);
      }
  }
else if (sdy.inputImageFormat == INPUT_NON_HEADER) {
  if (sdy.outputImageFormat != OUTPUT_NON_HEADER ||
        (sdy.subFName != SUB_FNAME_FIX &&
         sdy.subFName != SUB_FNAME_COMMAND_LINE)) {
      printf("Wrong InputImageFormat\n"); exit(0);
      }
  }
}

/*==================================================================
Get Current Date and Time as integer
OUTPUT      NumTime defined in TempSub.H
11/20/98 Code by Shige
==================================================================*/
#define     SIZET 20
NumTime GetNumTime()
{
NumTime     numTime;

time_t      timer = time(NULL);
tm    *lct  = localtime(&timer);

char  year  [SIZET];
char    yearS [SIZET];
char  month [SIZET];
char  day   [SIZET];
char  hour  [SIZET];
char  minute[SIZET];
char    second[SIZET];

/*-----------------------------------------------------------------
      Get Time
-----------------------------------------------------------------*/
strftime(year,    SIZET, "%Y", lct);
strftime(yearS,   SIZET, "%y", lct);
strftime(month,   SIZET, "%m", lct);
strftime(day,     SIZET, "%d", lct);
strftime(hour,    SIZET, "%H", lct);
strftime(minute,  SIZET, "%M", lct);
strftime(second,  SIZET, "%S", lct);

/*-----------------------------------------------------------------
      Conversion to integer
-----------------------------------------------------------------*/
numTime.year   = atoi(year);
numTime.yearS  = atoi(yearS);
numTime.month  = atoi(month);
numTime.day    = atoi(day);
numTime.hour   = atoi(hour);
numTime.minute = atoi(minute);
numTime.second = atoi(second);
```

```
return numTime;
}

/*==================================================================
Get Current Date and Time as strings
OUTPUT      TexTime defined in TempSub.H
11/20/98 Code by Shige
==================================================================*/
TexTime GetTexTime()
{
TexTime     texTime;
time_t      timer = time(NULL);
tm    *lct  = localtime(&timer);

/*-----------------------------------------------------------------
      Get Time
-----------------------------------------------------------------*/
strftime(texTime.year,   SIZET, "%Y", lct);
strftime(texTime.yearS,  SIZET, "%y", lct);
strftime(texTime.month,  SIZET, "%m", lct);
strftime(texTime.day,    SIZET, "%d", lct);
strftime(texTime.hour,   SIZET, "%H", lct);
strftime(texTime.minute, SIZET, "%M", lct);
strftime(texTime.second, SIZET, "%S", lct);

return texTime;
}

/*=====================================================================
Function Name     getFileSize
Function    Get file size in bytes
Usage       int FileSize = getFileSize(char *FileName);
Coded       07/21/98 Shige
=====================================================================*/
int getFileSize(char *FileName)
{
int         FSize;
FILE        *fp;

if ((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n",FileName) ; exit(0);
  }
fseek(fp, 0, SEEK_END);
FSize = ftell(fp);
fclose(fp);

return (FSize);
}

/*=====================================================================
Function Name     getImageFormat
Function    Get Image Format
Usage       InputImageFormat getImageFormat(char*, int*, int*);
Coded       12/04/98 Shige
=====================================================================*/
#define     FLAG_LEN    512
InputImageFormat getImageFormat(char *FileName, int *mszx, int *mszy)
{
FILE              *fp;
InputImageFormat  inputImageFormat;
```

```
int                 FileSize;
FcrStand            fcr;
FcrDicom            dcm;
ThvDicom            thv;
char                *header;
short               *header_gen;

int                 i, j;
int                 n = 0;
char                auf[FLAG_LEN], buf[FLAG_LEN];
char                FcrStdFlag[] = "FUJI PHOTO FILM";
char                      DcmFlag[]     = "1.2.840.10008.5.1.4.1.1.1";
char                      UoehFlag[]    = "FUJI PHOTO FILM";    // Not used
now
char                      UCFlag1[]     = "FUJI CR";            // Not used
now
char                      UCFlag2[]     = "Fuji Photo Film";    // Not used
now
char                ThvFlag[]     = "Philips Medical Systems";

extern      FcrStand getFcrStandInfo(FILE*, char*);
extern      FcrDicom getFcrDicomInfo(FILE*, char*);
extern      ThvDicom getThvDicomInfo(FILE*, char*);

/*-----------------------------------------------------------------
      Read top part of image
-----------------------------------------------------------------*/
if ((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n",FileName) ; exit(0);
  }
fread(auf, sizeof(char), FLAG_LEN, fp);
fclose(fp);

for (i = 0; i < FLAG_LEN; i++) {
  if (auf[i] >= 0x20 && auf[i] <= 0x7A) buf[n++] = auf[i];
  }
buf[n] = 0;

/*-----------------------------------------------------------------
      Identification of Image Format
-----------------------------------------------------------------*/
if (0 != strstr(buf, DcmFlag)) {
  if (0 != strstr(buf, ThvFlag)) {
    inputImageFormat = INPUT_DICOM_THV_HEADER;
    header = new char[MAX_THV_HEADER_SIZE];
    if ((fp = fopen(FileName, "r") ) == 0) {
      printf("%s is missing \n",FileName) ; exit(0);
      }
    thv = getThvDicomInfo(fp, header);
    fclose(fp);
    *mszx = thv.Col;
    *mszy = thv.Row;
    delete header;
    }
  else {
    inputImageFormat = INPUT_DICOM_FCR_HEADER;
    header = new char[MAX_FCR_DCM_HEADER_SIZE];
    if ((fp = fopen(FileName, "r") ) == 0) {
      printf("%s is missing \n",FileName) ; exit(0);
      }
    dcm = getFcrDicomInfo(fp, header);
```

```
    fclose(fp);
    *mszx = dcm.Col;
    *mszy = dcm.Row;
    delete header;
    }
  }
else if (0 != strstr(buf, FcrStdFlag)) {
  inputImageFormat = INPUT_STANDARD_FCR_HEADER;
  header = new char[HEADER_SIZE];
  if ((fp = fopen(FileName, "r") ) == 0) {
    printf("%s is missing \n",FileName) ; exit(0);
    }
  fcr = getFcrStandInfo(fp, header);
  fclose(fp);
  *mszx = fcr.Col;
  *mszy = fcr.Row;
  delete header;
  }
else {
  FileSize = getFileSize(FileName);
  header_gen = new short[GEN_HEADER_SIZE];
  if ((fp = fopen(FileName, "r") ) == 0) {
    printf("%s is missing \n",FileName) ; exit(0);
    }
  #ifdef _SUN_SPARC_SOLARIS_1X_
    fread((char*)header_gen, sizeof(short), GEN_HEADER_SIZE, fp);
  #else
    fread(header_gen, sizeof(short), GEN_HEADER_SIZE, fp);
  #endif
  fclose(fp);
  *mszx = header_gen[0];
  *mszy = header_gen[1];
  if (FileSize == (int)pow((int)sqrt(FileSize / 2), 2) * 2) {
    *mszx = *mszy = (int)sqrt(FileSize / 2);
    inputImageFormat = INPUT_NON_HEADER;
    }
  else if (FileSize == JSRT_DB_SIZE) {
    *mszx = *mszy = 2048;
    inputImageFormat = INPUT_JSRT_DB;
    }
  else if (FileSize == (*mszx * *mszy + GEN_HEADER_SIZE) * 2) {
    inputImageFormat = INPUT_STANDARD_GEN_HEADER;
    }
  else {
    inputImageFormat = INPUT_OTHER;
    }
  delete header_gen;
  }

return inputImageFormat;
}


/*=====================================================================
Function Name     getGenStandInfo
Function    Get General Standard Header Information
Coded       02/24/99 Shige
=====================================================================*/
GenStand getGenStandInfo(FILE *fp)
{
GenStand    gen;
```

```
short       *header = new short[GEN_HEADER_SIZE];

#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)header, sizeof(short), GEN_HEADER_SIZE, fp);
#else
  fread(header, sizeof(short), GEN_HEADER_SIZE, fp);
#endif
gen.Col = header[0];
gen.Row = header[1];
gen.NBit = 10;

delete header;
return gen;
}

/*=================================================================
Function Name     getSFCRImage
Function    Get FCR Standard Header Image(Short)
Usage       void getSFCRImage(char*, short*, int, int);
Coded       12/04/98 Shige
=================================================================*/
void getSFCRImage(char *FileName, short *image, int col, int lin)
{
}

/*=================================================================
Function Name     getSDCMImage
Function    Get DCM Image(Short)
Usage       void getSDCMImage(char*, short*, int, int);
Coded       12/07/98 Shige
=================================================================*/
void getSDCMImage(char *FileName, short *image, int col, int lin)
{
FILE        *fp;
FcrDicom    fcr;
char        *FcrDcmHeader;
float       ratio;
int             mszx, mszy;
short       *org;
short       MaxGray = 1023;
int         i, j, ix, iy, jx, jy, k, l, u, v;
extern              FcrDicom getFcrDicomInfo(FILE*, char*);

/*--------------------------------------------
  Begin
--------------------------------------------*/
FcrDcmHeader = new char[MAX_FCR_DCM_HEADER_SIZE];

/*--------------------------------------------
    Open Original Image File
--------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
fcr   = getFcrDicomInfo(fp, FcrDcmHeader);
mszx  = fcr.Col;
mszy  = fcr.Row; if (mszy > FCR_WY) mszy = FCR_WY;
ratio = (float)mszx / (float)col;
org   = new short[mszx * mszy];

/*--------------------------------------------
```

```
  Identification of Matrix Size
--------------------------------------------*/
if (0 == strcmp(fcr.IPSize, HANSETU_DCM)) {
  printf("IP Size is 14 x 17\n");
  if (0 == strcmp(fcr.Dir,  V_FLIP_DCM) ||
      0 == strcmp(fcr.Dir, HV_FLIP_DCM))
      fseek(fp, sizeof(short) * FCR_WX * SKIP_HANSETU, SEEK_CUR);
  }

/*--------------------------------------------------------------
  Read Original Image with 10 bits gray scale
--------------------------------------------------------------*/
printf("DCM.FCR %s (%d, %d)->(%d, %d)\n", FileName,
                                mszx, mszy,
                                col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), mszx * mszy, fp);
#else
  fread(org, sizeof(short), mszx * mszy, fp);
#endif
fclose(fp);

/*----------------------------------------------------
    Copy
-----------------------------------------------------*/
if (0 == strcmp(fcr.Dir, HV_FLIP_DCM)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = lin - j - 1;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = col - i - 1;
    u  = jy * col  + jx;
    v  = iy * mszx + ix;
    image[u] = ((org[v] >> 8 & 0x00FF) | (org[v] << 8)) ^ MaxGray;
    }
    }
  }
else if (0 == strcmp(fcr.Dir, H_FLIP_DCM)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = j;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = col - i - 1;
    u  = jy * col  + jx;
    v  = iy * mszx + ix;
    image[u] = ((org[v] >> 8 & 0x00FF) | (org[v] << 8)) ^ MaxGray;
    }
    }
  }
else if (0 == strcmp(fcr.Dir, V_FLIP_DCM)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = lin - j - 1;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = i;
    u  = jy * col  + jx;
    v  = iy * mszx + ix;
    image[u] = ((org[v] >> 8 & 0x00FF) | (org[v] << 8)) ^ MaxGray;
```

```
    }
    }
  }
else {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    u  = j  * col  + i;
    v  = iy * mszx + ix;
    image[u] = ((org[v] >> 8 & 0x00FF) | (org[v] << 8)) ^ MaxGray;
    }
    }
  }
delete org;
delete FcrDcmHeader;
}

/*=================================================================
Function Name     getSTHVImage
Function    Get Thoravision Image(Short)
Usage       void getSTHVImage(char*, short*, int, int);
Coded       12/07/98 Shige
=================================================================*/
void getSTHVImage(char *FileName, short *image, int col, int lin)
{
FILE        *fp;
ThvDicom    thv;
char        *THVHeader;
float       ratio;
short       *org;
short       MaxGray = 1023;
short       val;
int         Shift;
int         i, j, ix, iy, jx, jy, k, l, u, v;
extern              ThvDicom getThvDicomInfo(FILE*, char*);

/*--------------------------------------------
  Begin
--------------------------------------------*/
THVHeader = new char[MAX_THV_HEADER_SIZE];

/*--------------------------------------------
    Open Original Image File
--------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
thv   = getThvDicomInfo(fp, THVHeader);
ratio = (float)thv.Col / (float)col;
org   = new short[thv.Col * thv.Row];
Shift = thv.NBit - 10;

/*------------------------------------------------------------
  Read Original Image with 10 bits gray scale
------------------------------------------------------------*/
printf("PHL.THV %s (%d, %d)->(%d, %d)\n", FileName,
                                thv.Col, thv.Row,
                                col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), thv.Col * thv.Row, fp);
```

```
#else
  fread(org, sizeof(short), thv.Col * thv.Row, fp);
#endif
fclose(fp);

/*----------------------------------------------------
    Copy
-----------------------------------------------------*/
if (0 == strcmp(thv.Dir, AP_THV)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = j;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = col - i - 1;
    u  = jy * col    + jx;
    v  = iy * thv.Col + ix;
    if (ix >= thv.Col || iy >= thv.Row) image[u] = 0;
    else {
      val      = (org[v] >> 8 & 0x00FF) | (org[v] << 8 & 0xFF00);
      image[u] = (val >> Shift) ^ MaxGray;
      }
    }
    }
  }
else {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    u  = j * col    + i;
    v  = iy * thv.Col + ix;
    if (ix >= thv.Col || iy >= thv.Row) image[u] = 0;
    else {
      val      = (org[v] >> 8 & 0x00FF) | (org[v] << 8 & 0xFF00);
      image[u] = (val >> Shift) ^ MaxGray;
      }
    }
    }
  }
delete org;
delete THVHeader;
}

/*=================================================================
Function Name     getSNOHImage
Function    Get Non-Header Image(Short)
Usage       void getSNOHImage(char*, int, int, short*, int, int);
Coded       12/07/98 Shige
=================================================================*/
void getSNOHImage(char *FileName, int mx, int my,
              short *image, int col, int lin)
{
FILE        *fp;
float       ratio;
short       *org;
int         i, j, ix, iy, jx, jy, k, l, u, v;

/*--------------------------------------------
  Begin
--------------------------------------------*/
```

```
ratio = (float)mx / (float)col;
org   = new short[mx * my];

/*---------------------------------------------
    Open Original Image File
---------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
printf("NOH.IMG %s (%d, %d)->(%d, %d)\n", FileName,
                                mx, my,
                                col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), mx * my, fp);
#else
  fread(org, sizeof(short), mx * my, fp);
#endif
fclose(fp);

/*----------------------------------------------------
  Copy
----------------------------------------------------*/
for (j = 0; j < lin; j++) {
  iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
      ix = (float)i * ratio + 0.5;
      v= iy * mx + ix;
      u= j* col + i;
      image[u] = org[v];
      }
  }
delete org;
}

/*==============================================================
Function Name     getSJDBImage
Function    Get JSRT DB Image(Short)
Usage       void getSJDBImage(char*, int, int, short*, int, int);
Coded       12/07/98 Shige
==============================================================*/
void getSJDBImage(char *FileName, int mx, int my,
              short *image, int col, int lin)
{
FILE        *fp;
float       ratio;
short       *org;
short       MaxGray = 1023;
int         i, j, ix, iy, jx, jy, k, l, u, v;

/*---------------------------------------------
  Begin
---------------------------------------------*/
ratio = (float)mx / (float)col;
org   = new short[mx * my];

/*---------------------------------------------
    Open Original Image File
---------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
```

```
printf("JSRT.DB %s (%d, %d)->(%d, %d)\n", FileName,
                                mx, my,
                                col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), mx * my, fp);
#else
  fread(org, sizeof(short), mx * my, fp);
#endif
fclose(fp);

/*----------------------------------------------------
  Copy
----------------------------------------------------*/
for (j = 0; j < lin; j++) {
  iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
      ix = (float)i * ratio + 0.5;
      v= iy * mx + ix;
      u= j* col + i;
      image[u] = org[v] ^ MaxGray;
      }
  }
delete org;
}

/*==============================================================
Function Name     getSGENImage
Function    Get General Standard Header Image(Short)
Usage       void getSGENImage(char*, short*, int, int);
Coded       02/24/99 Shige
==============================================================*/
void getSGENImage(char *FileName, short *image, int col, int lin)
{
GenStand    gen;
FILE        *fp;
float       ratio;
short       *org;
int         i, j, ix, iy, jx, jy, k, l, u, v;

/*---------------------------------------------
    Open Original Image File
---------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
gen   = getGenStandInfo(fp);
ratio = (float)gen.Col / (float)col;
org   = new short[gen.Col * gen.Row];

printf("GEN.IMG %s (%d, %d)->(%d, %d)\n", FileName,
                                gen.Col, gen.Row,
                                col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), gen.Col * gen.Row, fp);
#else
  fread(org, sizeof(short), gen.Col * gen.Row, fp);
#endif
fclose(fp);

/*----------------------------------------------------
  Copy
```

```
----------------------------------------------------*/
for (j = 0; j < lin; j++) {
  iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
      ix = (float)i * ratio + 0.5;
      v= iy * gen.Col + ix;
      u= j* col + i;
      image[u] = org[v];
      }
  }
delete org;
}

/*==============================================================
Function Name     getCFCRImage
Function    Get FCR Standard Header Image(unsigned char)
Usage       void getCFCRImage(char*, unsigned char*, int, int);
Coded       12/04/98 Shige
==============================================================*/
void getCFCRImage(char *FileName, unsigned char *image, int col, int
lin)
{
}

/*==============================================================
Function Name     getCDCMImage
Function    Get DCM Image(unsigned char)
Usage       void getCDCMImage(char*, unsigned char*, int, int);
Coded       12/07/98 Shige
==============================================================*/
void getCDCMImage(char *FileName, unsigned char *image, int col, int
lin)
{
FILE        *fp;
FcrDicom    fcr;
char            *FcrDcmHeader;
float       ratio;
int         mszx, mszy;
short       *org;
unsigned char     MaxGray = 255;
int         Shift   = 2;
int         i, j, ix, iy, jx, jy, k, l, u, v;
extern            FcrDicom getFcrDicomInfo(FILE*, char*);

/*---------------------------------------------
  Begin
---------------------------------------------*/
FcrDcmHeader = new char[MAX_FCR_DCM_HEADER_SIZE];

/*---------------------------------------------
    Open Original Image File
---------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
fcr   = getFcrDicomInfo(fp, FcrDcmHeader);
mszx  = fcr.Col;
mszy  = fcr.Row; if (mszy > FCR_WY) mszy = FCR_WY;
ratio = (float)mszx / (float)col;
org   = new short[mszx * mszy];
```

```
/*---------------------------------------------
  Identification of Matrix Size
---------------------------------------------*/
if (0 == strcmp(fcr.IPSize, HANSETU_DCM)) {
  printf("IP Size is 14 x 17\n");
  if (0 == strcmp(fcr.Dir,  V_FLIP_DCM) ||
      0 == strcmp(fcr.Dir, HV_FLIP_DCM))
      fseek(fp, sizeof(short) * FCR_WX * SKIP_HANSETU, SEEK_CUR);
  }

/*--------------------------------------------------------------
  Read Original Image with 10 bits gray scale
--------------------------------------------------------------*/
printf("DCM.FCR %s (%d, %d)->(%d, %d)\n", FileName,
                                mszx, mszy,
                                col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), mszx * mszy, fp);
#else
  fread(org, sizeof(short), mszx * mszy, fp);
#endif
fclose(fp);

/*----------------------------------------------------
    Copy
----------------------------------------------------*/
if (0 == strcmp(fcr.Dir, HV_FLIP_DCM)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = lin - j - 1;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = col - i - 1;
    u  = jy * col  + jx;
    v  = iy * mszx + ix;
    image[u] = (((org[v] >> 8 & 0x00FF) |
             (org[v] << 8)) >> Shift) ^ MaxGray;
    }
    }
  }
else if (0 == strcmp(fcr.Dir, H_FLIP_DCM)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = j;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = col - i - 1;
    u  = jy * col  + jx;
    v  = iy * mszx + ix;
    image[u] = (((org[v] >> 8 & 0x00FF) |
             (org[v] << 8)) >> Shift) ^ MaxGray;
    }
    }
  }
else if (0 == strcmp(fcr.Dir, V_FLIP_DCM)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = lin - j - 1;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = i;
    u  = jy * col  + jx;
    v  = iy * mszx + ix;
    image[u] = (((org[v] >> 8 & 0x00FF) |
             (org[v] << 8)) >> Shift) ^ MaxGray;
    }
    }
  }
else {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    u  = j  * col  + i;
    v  = iy * mszx + ix;
    image[u] = (((org[v] >> 8 & 0x00FF) |
             (org[v] << 8)) >> Shift) ^ MaxGray;
    }
    }
  }
delete org;
delete FcrDcmHeader;
}

/*==================================================================
Function Name     getCTHVImage
Function    Get THV Image(unsigned char)
Usage       void getCTHVImage(char*, unsigned char*, int, int);
Coded       12/07/98 Shige
==================================================================*/
void getCTHVImage(char *FileName, unsigned char *image, int col, int
lin)
{
FILE        *fp;
ThvDicom    thv;
char            *THVHeader;
float       ratio;
short       *org;
unsigned char     MaxGray = 255;
short           val;
int         Shift;
int         i, j, ix, iy, jx, jy, k, l, u, v;
extern            ThvDicom getThvDicomInfo(FILE*, char*);

/*-------------------------------------------
  Begin
-------------------------------------------*/
THVHeader = new char[MAX_THV_HEADER_SIZE];

/*-------------------------------------------
    Open Original Image File
-------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
thv   = getThvDicomInfo(fp, THVHeader);
ratio = (float)thv.Col / (float)col;
org   = new short[thv.Col * thv.Row];
Shift = thv.NBit - 8;

/*--------------------------------------------------------------
  Read Original Image with 10 bits gray scale
--------------------------------------------------------------*/
printf("PHL.THV %s (%d, %d)->(%d, %d)\n", FileName,
                                thv.Col, thv.Row,
                                col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), thv.Col * thv.Row, fp);
#else
  fread(org, sizeof(short), thv.Col * thv.Row, fp);
#endif
fclose(fp);

/*----------------------------------------------------
    Copy
----------------------------------------------------*/
if (0 == strcmp(thv.Dir, AP_THV)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = j;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = col - i - 1;
    u  = jy * col   + jx;
    v  = iy * thv.Col + ix;
    if (ix >= thv.Col || iy >= thv.Row) image[u] = 0;
    else {
      val      = (org[v] >> 8 & 0x00FF) | (org[v] << 8 & 0xFF00);
      image[u] = (val >> Shift) ^ MaxGray;
      }
    }
    }
  }
else {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    u  = j * col   + i;
    v  = iy * thv.Col + ix;
    if (ix >= thv.Col || iy >= thv.Row) image[u] = 0;
    else {
      val      = (org[v] >> 8 & 0x00FF) | (org[v] << 8 & 0xFF00);
      image[u] = (val >> Shift) ^ MaxGray;
      }
    }
    }
  }
delete org;
delete THVHeader;
}

/*==================================================================
Function Name     getCNOHImage
Function    Get Non-Header Image(unsigned char)
Usage       void getCNOHImage(char*, int, int, unsigned char*, int, int)
Coded       12/07/98 Shige
==================================================================*/
void getCNOHImage(char *FileName, int mx, int my,
              unsigned char *image, int col, int lin)
{
FILE        *fp;
float       ratio;
short       *org;
int         Shift = 2;
int         i, j, ix, iy, jx, jy, k, l, u, v;

/*-------------------------------------------
  Begin
-------------------------------------------*/
ratio = (float)mx / (float)col;
org   = new short[mx * my];

/*-------------------------------------------
    Open Original Image File
-------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
printf("NOH.IMG %s (%d, %d)->(%d, %d)\n", FileName,
                                mx, my,
                                col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), mx * my, fp);
#else
  fread(org, sizeof(short), mx * my, fp);
#endif
fclose(fp);

/*----------------------------------------------------
  Copy
----------------------------------------------------*/
for (j = 0; j < lin; j++) {
  iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
      ix = (float)i * ratio + 0.5;
      v= iy * mx + ix;
      u= j* col + i;
      image[u] = org[v] >> Shift;
      }
  }
delete org;
}

/*==================================================================
Function Name     getCJDBImage
Function    Get JSRT DB Image(unsigned char)
Usage       void getCJDBImage(char*, int, int, unsigned char*, int, int)
Coded       12/07/98 Shige
==================================================================*/
void getCJDBImage(char *FileName, int mx, int my,
              unsigned char *image, int col, int lin)
{
FILE        *fp;
float       ratio;
short       *org;
unsigned char     MaxGray = 255;
int         Shift   = 2;
int         i, j, ix, iy, jx, jy, k, l, u, v;

/*-------------------------------------------
  Begin
-------------------------------------------*/
ratio = (float)mx / (float)col;
org   = new short[mx * my];

/*-------------------------------------------
    Open Original Image File
-------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
printf("JSRT.DB %s (%d, %d)->(%d, %d)\n", FileName,
                                mx, my,
                                col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), mx * my, fp);
#else
  fread(org, sizeof(short), mx * my, fp);
#endif
fclose(fp);

/*----------------------------------------------------
  Copy
----------------------------------------------------*/
for (j = 0; j < lin; j++) {
  iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
      ix = (float)i * ratio + 0.5;
      v= iy * mx + ix;
      u= j* col + i;
      image[u] = (org[v] >> Shift) ^ MaxGray;
      }
  }
delete org;
}

/*==================================================================
Function Name     getCGENImage
Function    Get General Standard Header Image(unsigned char)
Usage       void getCGENImage(char*, unsigned char*, int, int);
Coded       02/24/99 Shige
==================================================================*/
void getCGENImage(char *FileName, unsigned char *image, int col, int
lin)
{
GenStand    gen;
FILE        *fp;
float       ratio;
short       *org;
int         Shift = 2;
int         i, j, ix, iy, jx, jy, k, l, u, v;

/*-------------------------------------------
    Open Original Image File
-------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
gen   = getGenStandInfo(fp);
ratio = (float)gen.Col / (float)col;
org   = new short[gen.Col * gen.Row];

printf("GEN.IMG %s (%d, %d)->(%d, %d)\n", FileName,
                                gen.Col, gen.Row,
                                  col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), gen.Col * gen.Row, fp);
#else
  fread(org, sizeof(short), gen.Col * gen.Row, fp);
#endif
fclose(fp);

/*-------------------------------------------------------
  Copy
--------------------------------------------------------*/
for (j = 0; j < lin; j++) {
  iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
      ix = (float)i * ratio + 0.5;
      v= iy * gen.Col + ix;
      u= j* col + i;
      image[u] = org[v] >> Shift;
      }
  }
delete org;
}

/*==================================================================
Function Name     getODCMImage
Function    Get Reduced Original FCR DCM Image(Short)
Usage       void getODCMImage(char*, short*, int, int);
Coded       04/14/99 Shige
===================================================================*/
void getODCMImage(char *FileName, short *image, int col, int lin)
{
FILE        *fp;
FcrDicom    fcr;
char        *FcrDcmHeader;
float       ratio;
int             mszx, mszy;
short       *org;
int         i, j, ix, iy, jx, jy, k, l, u, v;
extern            FcrDicom getFcrDicomInfo(FILE*, char*);

/*--------------------------------------------
  Begin
--------------------------------------------*/
FcrDcmHeader = new char[MAX_FCR_DCM_HEADER_SIZE];

/*--------------------------------------------
    Open Original Image File
--------------------------------------------*/
if((fp = fopen(FileName, "r") ) == 0) {
  printf("%s is missing \n", FileName) ; exit(0);
  }
fcr   = getFcrDicomInfo(fp, FcrDcmHeader);
mszx  = fcr.Col;
mszy  = fcr.Row; if (mszy > FCR_WY) mszy = FCR_WY;
if (mszx == FCR_WX && col == FCR_WX / RATIO) ratio = (float)RATIO;
else                                    ratio = (float)mszx / (float)col;
org   = new short[mszx * mszy];

/*--------------------------------------------
  Identification of Matrix Size
--------------------------------------------*/
```

```
if (0 == strcmp(fcr.IPSize, HANSETU_DCM)) {
  printf("IP Size is 14 x 17\n");
  if (0 == strcmp(fcr.Dir,  V_FLIP_DCM) ||
      0 == strcmp(fcr.Dir, HV_FLIP_DCM))
      fseek(fp, sizeof(short) * FCR_WX * SKIP_HANSETU, SEEK_CUR);
  }

/*---------------------------------------------------------------
  Read Original Image with 10 bits gray scale
---------------------------------------------------------------*/
printf("DCM.FCR %s (%d, %d)->(%d, %d)\n", FileName,
                                 mszx, mszy,
                                 col, lin);
#ifdef _SUN_SPARC_SOLARIS_1X_
  fread((char*)org, sizeof(short), mszx * mszy, fp);
#else
  fread(org, sizeof(short), mszx * mszy, fp);
#endif
fclose(fp);

/*-----------------------------------------------------
    Copy
-------------------------------------------------------*/
if (0 == strcmp(fcr.Dir, HV_FLIP_DCM)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = lin - j - 1;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = col - i - 1;
    u  = jy * col  + jx;
    v  = iy * mszx + ix;
    image[u] = org[v];
    }
    }
  }
else if (0 == strcmp(fcr.Dir, H_FLIP_DCM)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = j;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = col - i - 1;
    u  = jy * col  + jx;
    v  = iy * mszx + ix;
    image[u] = org[v];
    }
    }
  }
else if (0 == strcmp(fcr.Dir, V_FLIP_DCM)) {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
    jy = lin - j - 1;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    jx = i;
    u  = jy * col  + jx;
    v  = iy * mszx + ix;
    image[u] = org[v];
    }
    }
```

```
  }
else {
  for (j = 0; j < lin; j++) {
    iy = (float)j * ratio + 0.5;
  for (i = 0; i < col; i++) {
    ix = (float)i * ratio + 0.5;
    u  = j  * col  + i;
    v  = iy * mszx + ix;
    image[u] = org[v];
    }
    }
  }
delete org;
delete FcrDcmHeader;
}
```

```
/*=======================================================================
      Main Program of Temporal Subtraction for QUICK Mode
      Feb. 20, 1997
========================================================================*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "TempSub.H"

#define     LEN 128

#ifdef KRL_GNU
  #define cts_quick_y3 cts_quick_y3__
#else
  #define cts_quick_y3 cts_quick_y3_
#endif

extern "C" void cts_quick_y3(char*, char*, char*, char*);

int main(int argc, char *argv[])
{
  FILE             *fp;
  StudyPara studyPara;
  char             DefFile[LEN];
  char             PreImage[LEN], CurImage[LEN], SubImage[LEN];
  char             Success[20],  Failure[20];
  extern    void    CheckDefFile(StudyPara);
  extern    StudyPara ReadTSubDefFile(char*);

  /*-------------------------------------------------------------
      Check Arguments
  -------------------------------------------------------------*/
  if (argc != 4 && argc != 5 && !(argc == 2 && (strcmp(argv[1],"--ver")
== 0))) {
    printf("\7Usage: %s DefFile PreImage CurImage [SubImage]\n", argv
[0]);
    exit(0);
    }

  if (argc == 2 && (strcmp(argv[1],"--ver") == 0))
    {
    printf("%s\n", TS_PROG_VER);
    exit(0);
    }

  strcpy(DefFile, argv[1]);
  studyPara = ReadTSubDefFile(DefFile);
  CheckDefFile(studyPara);

  if    (argc == 4 && (studyPara.subFName == SUB_FNAME_COMMAND_LINE)) {
    printf("Wrong SubFName in %s.\n", DefFile); exit(0);
    }
  else if (argc == 5 && (studyPara.subFName != SUB_FNAME_COMMAND_LINE))
{
    printf("Wrong SubFName in %s.\n", DefFile); exit(0);
    }

  /*-------------------------------------------------------------
      BEGIN
  -------------------------------------------------------------*/
  strcpy(Success, "Success");
strcpy(Failure, "Failure");

/*----------------------------------------------------------
    Transfer Arguments
------------------------------------------------------------*/
strcpy(PreImage, argv[2]);
strcpy(CurImage, argv[3]);
if (argc == 5) strcpy(SubImage, argv[4]);
printf("START...PreImage:\t%s\n", PreImage);
printf("        CurImage:\t%s\n", CurImage);

/*--------------------------------------------------------
    FLAG.LOG for PROGRAM START
----------------------------------------------------------*/
if ((fp = fopen("FLAG.LOG","w")) == NULL) {
  printf("FLAG.LOG can't be opened\n"); exit(0);
  }
if (studyPara.subFName != SUB_FNAME_COMMAND_LINE)
  fprintf(fp, "%s\t%s\tNoSubtractionImage\t%s\n", PreImage,
                                      CurImage,
                                      Failure);
else
  fprintf(fp, "%s\t%s\t%s\t%s\n", PreImage,
                          CurImage,
                          SubImage,
                          Failure);
fclose(fp);

/*--------------------------------------------------------
    TEMPORAL SUBTRACTION FOR FCR IMAGE
----------------------------------------------------------*/
cts_quick_y3(DefFile, PreImage, CurImage, SubImage);

/*--------------------------------------------------------
    FLAG.LOG for PROGRAM END
----------------------------------------------------------*/
if ((fp = fopen("FLAG.LOG","w")) == NULL) {
  printf("FLAG.LOG can't be opened\n"); exit(0);
  }
fprintf(fp, "%s\t%s\t%s\t%s\n", PreImage,
                        CurImage,
                        SubImage,
                        Success);
fclose(fp);

system("rm -f *.dat fort.*");
}
```

```
!==============================================================================
        subroutine Warp_and_Subtraction( image1, image2, image3, blank,
        1  ncol, nlin, FX, FY, grayscale, maxpv ,offset, image1rw1,magnify)
!==============================================================================
!       Ver. 1.0
!       Written by Akiko Kano,  Mar.29, 1993
!------------------------------------------------------------------------------
!       Function of the subroutine is warp of the image1 and then make
!       subtraction image.
!       This subroutine can be used for both first warping and second warping.
!       The subtraction image data is stored as Image3.
!
!       (1) Pixel values on the warped image are determined by a linear inter-
!           polation method.
!       (2) Offset value is given at subtraction so that the average pixel
!           value of the subtraction image is (maxpv/2).
!       (3) If "grayscale" is 1, which means that "0" corresponds to low optical
!           density, Image2 is subtracted from warped Image1.
!       (4) Pixels with no image data from Image1, due to warping or rotation,
!           will given (maxpv/2) on the subtraction image.
!       (5) The warped current image for the second warping is obtained by this
!           subroutine.
!       (6) The contrast of a subtraction-image can be magnified by a factor of
!           "magnify".
!==============================================================================
!       ARGUMENTS
!------------------------------------------------------------------------------
        implicit  none
c       integer*4 Warp_and_Subtraction
        integer*4 ncol, nlin              ! Matrix Size of Image1 and 2      [I]
        integer*2 image1(ncol,nlin)       ! Current Image Data               [I]
        integer*2 image2(ncol,nlin)       ! Previous Image Data              [I]
        integer*2 image3(ncol,nlin)       ! Subtraction Image Data           [O]
        integer*2 blank(ncol,nlin)        ! 1 -> Pixel with No Image Data    [I]
                                          ! 0 -> Pixel with Image Data
        real*4    FX(ncol,nlin)           ! Matrix of Fitted Shift Value DX  [I]
        real*4    FY(ncol,nlin)           ! Matrix of Fitted Shift Value DY  [I]
        integer*4 grayscale               ! 1:0->Lighter, -1:0->Darker       [I]
        integer*4 maxpv                   ! Maximum Pixel Value              [I]
        integer*4 offset                  ! Offset value for subtraction image[I]
        integer*2 image1rw1(ncol,nlin)    ! Warped Current Image Data        [O]
        real*4    magnify                 ! Contrast magnification factor    [I]
                                          ! Default = 2.0
!------------------------------------------------------------------------------
!       VARIABLES
!------------------------------------------------------------------------------
c       character str*80
        integer*2 warp1
        integer*4 ave1, ave2
        integer*4 C, L, C0, L0, intp
        real*4    X, Y, DX, DY
        real*4    I00, I01, I10, I11
c       integer*4 leng
        integer*4 id
        common    /LOGFILE/ id
!------------------------------------------------------------------------------
!       FUNCTIONS
!------------------------------------------------------------------------------
c       integer*4 Calculate_Average
c       integer*4 PutOutputF, UTL$STR_PRINT, UTL$FILE_WRITE
!------------------------------------------------------------------------------
♀
!------------------------------------------------------------------------------
```

```
!       1. DETERMINE OFFSET
!------------------------------------------------------------------------------
        call Calculate_Average( image1, ncol, nlin, ave1 )
        call Calculate_Average( image2, ncol, nlin, ave2 )
        write (*,*) 'Average 1 is ', ave1
        write (*,*) 'Average 2 is ', ave2
        offset = maxpv / 2 - ave1 + ave2
c       leng = UTL$STR_PRINT( str, ' Offset : %i', offset )
c       call PutOutputF( '%s', str )
c       call UTL$FILE_WRITE( id, str )

        write(id,*) ' Offset : ', offset
!------------------------------------------------------------------------------
!       2. GET WARPED IMAGE DATA
!------------------------------------------------------------------------------
        do L = 1, nlin
            do C = 1, ncol
!------------------------------------------------------------------------------
!                       2-1. DETERMINE SHIFTED LOCATION
!------------------------------------------------------------------------------
                        X = float(C) + FX(C,L)
                        Y = float(L) + FY(C,L)
!------------------------------------------------------------------------------
!                       2-2. SHIFTED LOCATION IS INSIDE IMAGE
!                               -> DETERMINE PIXEL VALUE BY LINEAR INTERPOLATION
!                       image1rw1(C,L): warped current image for the second warping.
!------------------------------------------------------------------------------
                        if ( X.ge.1.0 .and. X.le.float(ncol) .and.
        1                         Y.ge.1.0 .and. Y.le.float(nlin) ) then
                            C0 = int(X)
                            L0 = int(Y)

                            if ( blank(C0,L0).eq.0 .and. blank(C0+1,L0).eq.0 .and.
        1                        blank(C0,L0+1).eq.0 .and. blank(C0+1,L0+1).eq.0 ) then
                                DX = X - float(C0)
                                DY = Y - float(L0)
                                I00 = float( image1(C0,L0) )
                                I10 = float( image1(C0+1,L0) )
                                I01 = float( image1(C0,L0+1) )
                                I11 = float( image1(C0+1,L0+1) )

                                intp = nint( (1.0-DX)*(1.0-DY)*I00 + DX*(1.0-DY)*I10
        1                                         + (1.0-DX)*DY*I01 + DX*DY*I11 )

                                warp1 = max( 0, min( maxpv, intp ) )
                                image1rw1(C,L)=warp1
!------------------------------------------------------------------------------
!                           2-3. NO IMAGE DATA DUE TO ROTATION
!                                       -> SUBSTITUTE PIXEL VALUE WITH THAT OF IMAGE2
!                       image1rw1(C,L): warped current image for the second warping.
!------------------------------------------------------------------------------
                            else
                                warp1 = image2(C,L)
                                image1rw1(C,L)=0
                            end if
!------------------------------------------------------------------------------
!                       2-4. SHIFTED LOCATION IS OUTSIDE IMAGE
!                               -> SUBSTITUTE PIXEL VALUE WITH THAT OF IMAGE2
!                       image1rw1(C,L): warped current image for the second warping.
!------------------------------------------------------------------------------
                        else
                            warp1 = image2(C,L)
                                image1rw1(C,L)=0
```

```
                        end if
!------------------------------------------------------------------------------
!                       3. GET SUBTRACTION IMAGE DATA
!                       magnify -> contrast magnification factor for subtraction image
!                       image1rw1(C,L): warped current image for the second warping.
!------------------------------------------------------------------------------
                        image3(C,L) = int(float(image2(C,L) - warp1)*magnify)+offset
                        if ( grayscale.eq.1 ) image3(C,L) = maxpv - image3(C,L)
                        image3(C,L) = max( 0, min( maxpv, image3(C,L) ) )
c                       image1rw1(C,L)=warp1

            end do
        end do
!------------------------------------------------------------------------------
!       END
!------------------------------------------------------------------------------
        return
        end
♀
!==============================================================================
        subroutine Calculate_Average( image, ncol, nlin, ave )
!==============================================================================
!       Ver. 1.0
!       Written by Akiko Kano,  Jan.14, 1993
!------------------------------------------------------------------------------
!       This function calculates the average pixel value of an image.
!==============================================================================
!       ARGUMENTS
!------------------------------------------------------------------------------
        implicit  none
c       integer*4 Calculate_Average
        integer*4 ncol, nlin              ! Matrix Size of Image Data        [I]
        integer*2 image(ncol,nlin)        ! Image Data                       [I]
        integer*4 ave                     ! Average Pixel Value              [O]
        real*8 sum
!------------------------------------------------------------------------------
!       VARIABLES
!------------------------------------------------------------------------------
        integer*4 C, L
!------------------------------------------------------------------------------
!       BEGIN
!------------------------------------------------------------------------------
        sum = 0
        do L = 1, nlin
            do C = 1, ncol
                sum = sum + image(C,L)
            end do
        end do
        ave = nint(sum / ( ncol * nlin ))
!------------------------------------------------------------------------------
!       END
!------------------------------------------------------------------------------
        return
        end
```

```
!==============================================================================
      subroutine warp_rib_chs(FX,FY,ncol, nlin,ribfeature1,
     &         rribcage1,lribcage1,rribcage_no1,lribcage_no1,
     &         ires_r_x,ires_r_y,ires_l_x,ires_l_y,num1,num2,
     &         iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2,
     &         ix1_rule3,image1_2)
!==============================================================================
!      In order to achive the lung segmentation of the warped current image,
!      we have to get the warped ribcage edges and cardioac edges of the current
!      image.
!      Function of the subroutine is to warp the detected cardiac edges and
!      detected ribcage edges for the segmentation of warped current image.
!      The warped current image is used for the second warping technique alone.
!------------------------------------------------------------------------------
!      ARGUMENTS
!------------------------------------------------------------------------------
       implicit  integer*4 (i-n)
       real*4    FX(ncol,nlin)          ! Matrix of Fitted Shift Value DX   [I]
       real*4    FY(ncol,nlin)          ! Matrix of Fitted Shift Value DY   [I]
       integer*4 ncol, nlin             ! Matrix Size of Image              [I]
       integer*2 ribfeature1(50)        ! Ribcage-Related Features for Image1 [I/O]
       integer*2 rribcage1(2,1215)      ! Right Ribcage Points for Image1   [I/O]
       integer*2 lribcage1(2,1215)      ! Left  Ribcage Points for Image1   [I/O]
       integer*2 rribcage_no1           ! No. of Right Ribcage Points for Image1 [I]
       integer*2 lribcage_no1           ! No. of Left  Ribcage Points for Image1 [I]
       integer*4 ires_r_x(100),ires_r_y(100) ! Detected location of R-cardiac edge
[I/O]
       integer*4 ires_l_x(100),ires_l_y(100) ! Detected location of L-cardiac edge
[I/O]
       integer*2 image1_2(ncol,nlin)    ! Image of Warped ribcage edge lines
                                        ! and cardiac edge lines.
                                        ! This is for program check.       [O]

       integer*2 iwt(586,586)           ! wark area for program check.
!------------------------------------------------------------------------------
       integer*4 ix1_rule2,iy1_rule2    ! Upper-right location of mediastinum
       integer*4 ix2_rule2,iy2_rule2    ! Upper-left of left cardiac edge
       integer*4 ix1_rule3(2,586)       ! x-locations of cardiac edges
                                        ! ix1_rule3(1,586) => right cardiac
x-location
                                        ! ix1_rule3(2,586) => left cardiac
x-location
       integer*4 iy1_rule1              ! bottom of right cardiac edge
!
!      (ix1_rule2,iy1_rule2) --------> x       |
!                                      |       |
!                                      |       |
!                                      |       |
!                                      |       |
!                                      |       x <-------- (ix2_rule2,iy2_rule2)
!                                     /         \
!                                     |          \
!                                     |            |
!               iy1_rule1   --------> x            |
!
!------------------------------------------------------------------------------
!------------------------------------------------------------------------------
!      ARGUMENTS
!------------------------------------------------------------------------------
       do j=1,586
         do i=1,586
              iwt(i,j)=0
```

```
         end do
       end do
!------------------------------------------------------------------------------
!      2. GET WARPED DATA
!------------------------------------------------------------------------------
       do L = 1, rribcage_no1
!------------------------------------------------------------------------------
!              2-1. DETERMINE SHIFTED LOCATIONS for right ribcage edge
!------------------------------------------------------------------------------
               X = float(rribcage1(1,L)) - FX(rribcage1(1,L),rribcage1(2,L))
               Y = float(rribcage1(2,L)) - FY(rribcage1(1,L),rribcage1(2,L))
               rribcage1(1,L)=int(X)
               rribcage1(2,L)=int(Y)
       end do
       do L = 1, lribcage_no1
!------------------------------------------------------------------------------
!              2-2. DETERMINE SHIFTED LOCATIONS for left ribcage edge
!------------------------------------------------------------------------------
               X = float(lribcage1(1,L)) - FX(lribcage1(1,L),lribcage1(2,L))
               Y = float(lribcage1(2,L)) - FY(lribcage1(1,L),lribcage1(2,L))
               lribcage1(1,L)=int(X)
               lribcage1(2,L)=int(Y)
       end do

!------------------------------------------------------------------------------
!              2-3. DETERMINE SHIFTED LOCATIONS for important ribcage features
!                       (top lung and midline)
!------------------------------------------------------------------------------
               Y = float(ribfeature1(1)) - FY(lribcage1(1,1),ribfeature1(1))
               ribfeature1(1) = int(Y)
               X = float(ribfeature1(6)) - FX(ribfeature1(3),ribfeature1(6))
               ribfeature1(6) = int(X)
               Y = float(ribfeature1(3)) - FY(ribfeature1(3),ribfeature1(6))
               ribfeature1(3) = int(Y)

!------------------------------------------------------------------------------
!              2-4. DETERMINE SHIFTED LOCATIONS for right cardiac edge
!------------------------------------------------------------------------------
       do L=1,num1
              X = float(ires_r_x(L)) - FX(ires_r_x(L),ires_r_y(L))
              Y = float(ires_r_y(L)) - FY(ires_r_x(L),ires_r_y(L))
              ires_r_x(L)=int(X)
              ires_r_y(L)=int(Y)
       end do
!------------------------------------------------------------------------------
!              2-5. DETERMINE SHIFTED LOCATIONS for left cardiac edge
!------------------------------------------------------------------------------
       do L=1,num2
              X = float(ires_l_x(L)) - FX(ires_l_x(L),ires_l_y(L))
              Y = float(ires_l_y(L)) - FY(ires_l_x(L),ires_l_y(L))
              ires_l_x(L)=int(X)
              ires_l_y(L)=int(Y)
       end do

!------------------------------------------------------------------------------
!              2-3. MAKE IMAGES FOR PROGRAM CHECK.
!              Warped ribcage edge lines and cardiac edge lines are
!              overlayed in the image1_2 and iwt.
!------------------------------------------------------------------------------
       do L=2,num1
              ix=ires_r_x(L)
              iy=ires_r_y(L)
              ix2=ires_r_x(L-1)
```

```
              iy2=ires_r_y(L-1)
          call line3(image1_2,ncol,nlin,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,ncol,nlin,ix,iy,ix2,iy2,ierr3)
       end do
       do L=2,num2
              ix=ires_l_x(L)
              iy=ires_l_y(L)
              ix2=ires_l_x(L-1)
              iy2=ires_l_y(L-1)
          call line3(image1_2,ncol,nlin,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,ncol,nlin,ix,iy,ix2,iy2,ierr3)
       end do
              ix=ires_r_x(1)
              iy=ires_r_y(1)
              ix2=ires_r_x(1)
              iy2=1
          call line3(image1_2,ncol,nlin,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,ncol,nlin,ix,iy,ix2,iy2,ierr3)
              ix=ires_l_x(1)
              iy=ires_l_y(1)
              ix2=ires_l_x(1)
              iy2=1
          call line3(image1_2,ncol,nlin,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,ncol,nlin,ix,iy,ix2,iy2,ierr3)
          if(ires_r_y(num1).gt.ires_l_y(num2)) then
              ix=ires_l_x(num2)
              iy=ires_l_y(num2)
              ix2=ires_l_x(num2)
              iy2=ires_r_y(num1)
          call line3(image1_2,ncol,nlin,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,ncol,nlin,ix,iy,ix2,iy2,ierr3)
           end if

           do j=1,nlin
               ix1_rule3(1,j)=0
               ix1_rule3(2,j)=0
           end do
           iy1_rule1=ires_r_y(num1)
           ix1_rule2=ires_r_x(1)
           iy1_rule2=1
           ix2_rule2=ires_l_x(1)
           iy2_rule2=max(ires_r_y(1),ires_l_y(1))
           do j=iy2_rule2,iy1_rule1
              do i=1,ncol
                 if(iwt(i,j).gt.0) then
                      ix1_rule3(1,j)=i
                      goto 7577
                 end if
              end do
 7577         do i2=ncol,1,-1
                 if(iwt(i2,j).gt.0) then
                      ix1_rule3(2,j)=i2
                      goto 7578
                 end if
              end do
 7578      end do

c         write(*,*) iy1_rule1,ix1_rule2,iy1_rule2,ix2_rule2,iy2_rule2
c           do j=iy2_rule2,iy1_rule1
c               write(*,*) j,ix1_rule3(1,j),ix1_rule3(2,j)
c           end do
              ix=1
              iy=ires_r_y(num1)
```

```
              ix2=ires_r_x(num1)
              iy2=ires_r_y(num1)
          call line3(image1_2,ncol,nlin,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,ncol,nlin,ix,iy,ix2,iy2,ierr3)
              ix=ires_l_x(num2)
              iy=ires_r_y(num1)
              ix2=ncol
              iy2=ires_r_y(num1)
          call line3(image1_2,ncol,nlin,ix,iy,ix2,iy2,ierr3)
          call line3(iwt,ncol,nlin,ix,iy,ix2,iy2,ierr3)
c        call writeimage(iwt,'chs.img',586,586)
!------------------------------------------------------------------------------
!      END
!------------------------------------------------------------------------------
       return
       end
```

```
c       =================================================================
c       SUBROUTINES FOR FITTING WITH 2-DIMENSIONAL N-TH ORDER POLYNOMIALS
c       WITH WEIGHTING
c       =================================================================
c       COMPILE : FOR/G_F WEIGHTED_FIT_INTP
c       =================================================================
        subroutine weighted_Fit_Prepare(npx,npy,n,weight)
c       DETERMINE MATRIX C AND INVERSE MATRIX D
c       (npx,npy)        MATRIX SIZE OF IMAGE                 (IN)
c       n                NO. OF ORDER (< 11)                  (IN)
c       weight(npx,npy) WEIGHTING FACTOR (>=0.0,<=1.0)        (IN)
c       ak               REDUCTION FACTOR (0.01)              (IN)
c       iwp(i,j)         i=1 X-LOCATION OF C
c                          2 Y-LOCATION OF C
c                          3 POWER OF x (X-LOCATION OF xp)
c                          4 POWER OF y (Y-LOCATION OF yp)
c                          j=1,...,nop
c       iwn(i,j)         i=1 X-LOCATION OF C
c                          2 Y-LOCATION OF C
c                        C(iwn(1,j),iwn(2,j)) = C(iwn(3,j),iwn(4,j))
c                        j=1,...,non
c       THIS SUBROUTINE CALLS INVERSED.
c       5/19/92  BY S.KATSURAGAWA (TRN2DC.F)
c       9/28/92   MODIFIED BY A.KANO
c       12/29/92  MODIFIED BY A.KANO
c       =================================================================
        implicit integer*4 (i-n)
        integer*2 ic(2,66,66),iwp(4,231),iwn(4,1980)
        real*4    weight(npx,npy)
        real*8    c(66,66),d(66,66),xp(21),yp(21),x,y,cc
        common    /trend/not,iwp,ak,d
c       -----------------------------------------------------------------
c       SET CONSTANTS
c       -----------------------------------------------------------------
c        do j=1,npy
c            weight(npx,j)=0.0
c        end do
        not =(n+1)*(n+2)/2
        n2p1=2*n+1
c       ak  =0.1
        ak  =0.01
c       -----------------------------------------------------------------
c       CLEAR BUFFER
c       -----------------------------------------------------------------
        non=0
        do 100 j=1,not
            do 101 i=j,not
                c(i,j)=0.0
101         continue
100     continue
c       -----------------------------------------------------------------
c       MAKE LOOKUP TABLE FOR THE POWER OF X AND Y
c       -----------------------------------------------------------------

        do 110 k=1,n+1
            do 111 m=1,k
                j=(k-1)*k/2+m
                do 112 is=n+1,1,-1
                    do 113 it=is,1,-1
                        i=(is-1)*is/2+it
                        if(i.lt.j) goto 111
                        ic(1,i,j)=is-it+k-m+1
                        ic(2,i,j)=it+m-1
```

```
113                 continue
112             continue
111         continue
110     continue
c       -----------------------------------------------------------------
c       SET INDICATOR TO BE CALCULATED ; ROW
c       -----------------------------------------------------------------
        do 120 i=1,not
            iwp(1,i)=i
            iwp(2,i)=1
            iwp(3,i)=ic(1,i,1)
            iwp(4,i)=ic(2,i,1)
120     continue
        nop=not
c       -----------------------------------------------------------------
c       SET INDICATOR TO BE CALCULATED ; COLUMN
c       -----------------------------------------------------------------
        do 121 i=2,not
            nop=nop+1
            iwp(1,nop)=not
            iwp(2,nop)=i
            iwp(3,nop)=ic(1,not,i)
            iwp(4,nop)=ic(2,not,i)
121     continue
c       -----------------------------------------------------------------
c       SET INDICATOR NOT TO BE CALCULATED
c       -----------------------------------------------------------------
        do 130 j=2,not-1
            do 131 i=j,not-1
                do 132 jq=1,j-1
                    do 133 iq=jq,not
                        if(ic(1,i,j).eq.ic(1,iq,jq).and.
     &                                  ic(2,i,j).eq.ic(2,iq,jq)) then
                            non=non+1
                            iwn(1,non)=i
                            iwn(2,non)=j
                            iwn(3,non)=iq
                            iwn(4,non)=jq
                            goto 131
                        end if
133                 continue
132             continue
                nop=nop+1
                iwp(1,nop)=i
                iwp(2,nop)=j
                iwp(3,nop)=ic(1,i,j)
                iwp(4,nop)=ic(2,i,j)
131         continue
130     continue
c       -----------------------------------------------------------------
c       CA=B.....A=(INVERSE C)*B
c       DETERMINATION OF MATRIX C
c       -----------------------------------------------------------------
        do 10 iy=1,npy
c           -------------------------------------------------------------
c           DETERMINATION OF y**k (k=0,1,....2*n)
c           -------------------------------------------------------------
            y=float(iy)*ak
            yp(1)=1.0
            do 11 i=2,n2p1
                yp(i)=y*yp(i-1)
11          continue
            do 20 ix=1,npx
```

```
                if ( weight(ix,iy).gt.0.0 ) then
c                 -------------------------------------------------------
c                 DETERMINATION OF x**k (k=0,1,....2*n)
c                 -------------------------------------------------------
                  x=float(ix)*ak
                  xp(1)=1.0
                  do 21 i=2,n2p1
                      xp(i)=x*xp(i-1)
21                continue
c                 -------------------------------------------------------
c                 DETERMINATION OF MATRIX C
c                 -------------------------------------------------------
                  do 40 i=1,nop
                      cc = weight(ix,iy)*xp(iwp(3,i))*yp(iwp(4,i))
                      c(iwp(1,i),iwp(2,i))=c(iwp(1,i),iwp(2,i))+ cc
40                continue
                end if
20          continue
10      continue
c       -----------------------------------------------------------------
c       TRANSFER MATRIX ELEMENTS WITH SAME VALUES
c       -----------------------------------------------------------------
        do 50 i=1,non
            c(iwn(1,i),iwn(2,i))=c(iwn(3,i),iwn(4,i))
50      continue
c       -----------------------------------------------------------------
c       SYMMETRIC MATRIX
c       -----------------------------------------------------------------
        do 52 j=2,not
            do 53 i=1,j-1
                c(i,j)=c(j,i)
53          continue
52      continue
c       -----------------------------------------------------------------
c       DETERMINATION OF INVERSE MATRIX D(I,J)
c       -----------------------------------------------------------------
        call inversed(c,d,66,not)
c
c       write(*,*)'NOP = ',nop,' NON = ',non
c
        return
        end
♀
c       =================================================================
        subroutine weighted_Fit_Intp(ibuf,xr,npx,npy,iwx,iwy,ps,dis,
     &                                 n,offset,weight,fitxy)

C       ibuf     INPUT INTEGER VALUES OF POINTS                      (IN)
C       xr       OUTPUT FITTED SURFACE(REAL NUMBER)                 (OUT)
C       (npx,npy)        BUFFER SIZE OF POINTS                       (IN)
C       (iwx,iwy)        MATRIX SIZE OF FITTED SURFACE               (IN)
C       ps               START POINT OF IBUF(1:X,2:Y)                (IN)
C       dis              DISTANCE BETWEEN INPUT POINTS(1:X,2:Y)      (IN)
C       n        ORDER OF POLYNOMIALS (N < 11)                       (IN)
C       offset   OFFSET VALUE OF IBUF                                (IN)
C       weight(npx,npy) WEIGHTING FACTOR (>=0.0, <= 1.0)
C       5/16/92  BY S.KATSURAGAWA (TRN2DN.F)
C       12/11/92   MODIFIED BY A.KANO
C       12/29/92   MODIFIED BY A.KANO
C       4/9/93   MODIFIED BY A.KANO
C       =================================================================
        implicit integer*4 (i-n)
        integer*4 ibuf(npx,npy)
```

```
        integer*4 fitxy(npx,npy)
        integer*4 ps(2), pe(2), dis(2)
        integer*4 offset
        integer*4 C, L, DC, DL, SC, SL, ix, iy
        integer*2 iwp(4,231)
        real*4    weight(npx,npy)
        real*4    xr(iwx,iwy)
        real*4    R00, R10, R01, R11, DRC, DRL
        real*8    a(66),b(66),d(66,66),xp(11),yp(11),x,y,z,bb
        common    /trend/not,iwp,ak,d
c       -----------------------------------------------------------------
c       CLEAR BUFFER
c       -----------------------------------------------------------------
c        do j=1,npy
c            weight(npx,j)=0.0
c        end do
        do 100 i=1,not
            b(i)=0.0
100     continue
        do iy=1,iwy
            do ix=1,iwx
                xr(ix,iy)=0.0
            end do
        end do
        do iy=1,npy
            do ix=1,npx
                fitxy(ix,iy)=0.0
            end do
        end do
c       -----------------------------------------------------------------
c       SUBTRACT OFFSET
c       -----------------------------------------------------------------
        do iy=1,npy
            do ix=1,npx
                ibuf(ix,iy)=ibuf(ix,iy)-offset
            end do
        end do
c       -----------------------------------------------------------------
c       CA=B....A=(INVERSE C)*B ;D(INVERSE C) IS OBTAINED PREVIOUSLY
c       DETERMINATION OF MATRIX B
c       -----------------------------------------------------------------
        do 10 iy=1,npy
c           -------------------------------------------------------------
c           DETERMINATION OF y**k (k=0,1,....n)
c           -------------------------------------------------------------
            y=float(iy)*ak
            yp(1)=1.0
            do 11 i=2,n+1
                yp(i)=y*yp(i-1)
11          continue
            do 20 ix=1,npx
              if ( weight(ix,iy).gt.0.0 ) then
c                 -------------------------------------------------------
c                 DETERMINATION OF x**k (k=0,1,....n)
c                 -------------------------------------------------------
                  x=float(ix)*ak
                  xp(1)=1.0
                  do 21 i=2,n+1
                      xp(i)=x*xp(i-1)
21                continue
c                 -------------------------------------------------------
c                 DETERMINATION OF Zij
c                 -------------------------------------------------------
```

```
                  z=float(ibuf(ix,iy))
c                 -------------------------------------------------------
c                 DETERMINATION OF MATRIX B
c                 -------------------------------------------------------
                  do 30 i=1,not
                      bb = weight(ix,iy)*xp(iwp(3,i))*yp(iwp(4,i))*z
                      b(i) = b(i) + bb
30                continue
                end if
20          continue
10      continue
c       -----------------------------------------------------------------
c       DETERMINATION OF MATRIX A(I)
c       -----------------------------------------------------------------
        do 60 i=1,not
                a(i)=0.0
                do 61 j=1,not
                        a(i)=a(i)+d(i,j)*b(j)
61              continue
60      continue

c       -----------------------------------------------------------------
c       FITTING
c       -----------------------------------------------------------------
        pe(1)=ps(1)+dis(1)*(npx-1)
        pe(2)=ps(2)+dis(2)*(npy-1)
c
        do 70 iy=1, npy
c           -------------------------------------------------------------
c           DETERMINATION OF y**k (k=0,1,....n)
c           -------------------------------------------------------------
            L=ps(2)+dis(2)*(iy-1)
            y=float(iy)*ak
            yp(1)=1.0
            do 71 i=2,n+1
                yp(i)=y*yp(i-1)
71          continue
            do 72 ix=1,npx
c               ---------------------------------------------------------
c               DETERMINATION OF x**k (k=0,1,....n)
c               ---------------------------------------------------------
                C=ps(1)+dis(1)*(ix-1)
                x=float(ix)*ak
                xp(1)=1.0
                do 73 i=2,n+1
                    xp(i)=x*xp(i-1)
73              continue
c
                z=0.0
                do 76 i=1,not
                    z=z+a(i)*xp(iwp(3,i))*yp(iwp(4,i))
76              continue
                xr(C,L)=z
72          continue
70      continue
c       -----------------------------------------------------------------
c       INTERPOLATION
c       -----------------------------------------------------------------
        do 80 iy=1,npy-1
c       do 80 iy=1,npy
            L=ps(2)+dis(2)*(iy-1)
            SL=L
            do 81 ix=1,npx-1
```

```
c           do 81 ix=1,npx
                C=ps(1)+dis(1)*(ix-1)
                SC=C
                R00=xr(SC,SL)
                R10=xr(SC+dis(1),SL)
                R01=xr(SC,SL+dis(2))
                R11=xr(SC+dis(1),SL+dis(2))

                L=SL
                DRC=(R10-R00)/float(dis(1))
                do 82 while (C.lt.(ps(1)+dis(1)*ix))
                    C=C+1
                    xr(C,SL)=xr(C-1,SL)+DRC
82              continue

                DRL=(R01-R00)/float(dis(2))
                do 83 while (L.lt.(ps(2)+dis(2)*iy))
                    L=L+1
                    C=SC
                    xr(SC,L)=xr(SC,L-1)+DRL

                    DL=L-SL
                    DRC=float(dis(2)-DL)*(R10-R00)+float(DL)*(R11-R01)
                    DRC=DRC/float(dis(1)*dis(2))
                    do 84 while (C.lt.(ps(1)+dis(1)*ix))
                        C=C+1
                        xr(C,L)=xr(C-1,L)+DRC
84                  continue
83              continue
                  fitxy(ix,iy)=int(xr(C,L))+offset
c                   write(*,*) ix,iy,c,l,fitxy(ix,iy)
81          continue
80      continue
c       -----------------------------------------------------------------
c       EXTRAPOLATION
c       -----------------------------------------------------------------
        do 85 L=1,ps(2)-1
            do 86 C=ps(1),pe(1)
                dl=ps(2)-L
                xr(C,L)=xr(C,ps(2))+float(dl)/float(dis(2))
     &                             *(xr(C,ps(2))-xr(C,ps(2)+dis(2)))
86          continue
85      continue
        do 90 L=pe(2)+1,iwy
            do 91 C=ps(1),pe(1)
                dl=L-pe(2)
                xr(C,L)=xr(C,pe(2))+float(dl)/float(dis(2))
     &                             *(xr(C,pe(2))-xr(C,pe(2)-dis(2)))
91          continue
90      continue

        do 92 L=1,iwy
            do 93 C=1,ps(1)-1
                dc=ps(1)-C
                xr(C,L)=xr(ps(1),L)+float(dc)/float(dis(1))
     &                             *(xr(ps(1),L)-xr(ps(1)+dis(1),L))
93          continue
            do 94 C=pe(1)+1,iwx
                dc=C-pe(1)
                xr(C,L)=xr(pe(1),L)+float(dc)/float(dis(1))
     &                             *(xr(pe(1),L)-xr(pe(1)-dis(1),L))
94          continue
92      continue
```

```
c       -----------------------------------------------------------------------
c       ADD OFFSET
c       -----------------------------------------------------------------------
        do L=1,iwy
             do C=1,iwx
                  xr(C,L)=xr(C,L)+float(offset)
             end do
        end do

        return
        end
```

```
c       ---------------------------------------------------------------
c       This subroutine can be simplified in future.
c       ---------------------------------------------------------------
c       ===============================================================
        subroutine Weighted_noFit_Intp(ibuf,xr,npx,npy,iwx,iwy,ps,dis,
     &                                 n,offset,weight,fitxy)
!       ---------------------------------------------------------------
!       Function of this subroutine is to determine the shift vectors
!       for the warping based on linear interpolation technique.
!
!       Interpolation part is originaly coded by A. Kano.
!       Extrapolation part is modified by T. Ishida.
!       For the extrapolation in this subroutine, shift vactors of the
!       boundary of lung-area rectangular are simply spreaded to the
!       outside.
!
c       ibuf    INPUT INTEGER VALUES OF POINTS                     (IN)
c       xr      OUTPUT SHIFT VALUES (REAL NUMBER)                  (OUT)
c       (npx,npy)       BUFFER SIZE OF POINTS                      (IN)
c       (iwx,iwy)       MATRIX SIZE OF FITTED SURFACE              (IN)
c       ps              START POINT OF IBUF(1:X,2:Y)               (IN)
c       dis             DISTANCE BETWEEN INPUT POINTS(1:X,2:Y)     (IN)
! ******************************************************************
!            Not used in this subroutine
c       n       ORDER OF POLYNOMIALS (N < 11)                      (IN)
! ******************************************************************
c       offset  OFFSET VALUE OF IBUF                               (IN)
! ******************************************************************
!            Not used in this subroutine
c       weight(npx,npy) WEIGHTING FACTOR (>=0.0, <= 1.0)
! ******************************************************************
c       fitxy(npx,npy)  Copy of Original shift values for program
c                       check.                                     (OUT)
c
c       4/9/93   ORIGINALY CODED BY A.KANO
c       1/12/98  CODED BY T. ISHIDA
c       ===============================================================
        implicit  integer*4 (i-n)
        integer*4 ibuf(npx,npy)         ! Original shift values
        integer*4 fitxy(npx,npy)        ! Copy of Original shift
                                        ! values for program check.
        integer*4 offset                ! OFFSET VALUE OF IBUF
        integer*4 ps(2), pe(2), dis(2)
        integer*4 C, L, DC, DL, SC, SL, ix, iy
! ******************************************************************
!            Not used in this subroutine
        real*4    weight(npx,npy)       ! WEIGHTING FACTOR
! ******************************************************************
        real*4    xr(iwx,iwy)           ! OUTPUT INTERPOLATED SHIFT VALUES
                                        ! Size are expanded to actual input
                                        ! image by using interpolation and
                                        ! extrapolation.
        real*4    R00, R10, R01, R11, DRC, DRL
        real*8    x,y,z
c       ---------------------------------------------------------------
c       CLEAR BUFFER
c       ---------------------------------------------------------------
        do iy=1,iwy
            do ix=1,iwx
                xr(ix,iy)=0.0
            end do
        end do
        do iy=1,npy
```

```
            do ix=1,npx
                fitxy(ix,iy)=0.0
            end do
        end do
c       ---------------------------------------------------------------
c       SUBTRACT OFFSET
c       ---------------------------------------------------------------
        do iy=1,npy
            do ix=1,npx
                ibuf(ix,iy)=ibuf(ix,iy)-offset
            end do
        end do
c       ---------------------------------------------------------------
c       LINEAR INTERPOLATION
|       Make shift vector maps of xr(C,L) and fitxy(ix,iy).
|       fitxy(ix,iy) is copy of original shift vector for program check.
c       ---------------------------------------------------------------
        pe(1)=ps(1)+dis(1)*(npx-1)
        pe(2)=ps(2)+dis(2)*(npy-1)
c
        do 70 iy=1, npy
            L=ps(2)+dis(2)*(iy-1)
            do 72 ix=1,npx
                C=ps(1)+dis(1)*(ix-1)
                xr(C,L)=float(ibuf(ix,iy))
                fitxy(ix,iy)=int(xr(C,L))+offset
72          continue
70      continue
c       ---------------------------------------------------------------
c       INTERPOLATION
c       ---------------------------------------------------------------
        do 80 iy=1,npy-1
            L=ps(2)+dis(2)*(iy-1)
            SL=L
            do 81 ix=1,npx-1
                C=ps(1)+dis(1)*(ix-1)
                SC=C
                R00=xr(SC,SL)
                R10=xr(SC+dis(1),SL)
                R01=xr(SC,SL+dis(2))
                R11=xr(SC+dis(1),SL+dis(2))

                L=SL
                DRC=(R10-R00)/float(dis(1))
                do 82 while (C.lt.(ps(1)+dis(1)*ix))
                    C=C+1
                    xr(C,SL)=xr(C-1,SL)+DRC
82              continue

                DRL=(R01-R00)/float(dis(2))
                do 83 while (L.lt.(ps(2)+dis(2)*iy))
                    L=L+1
                    C=SC
                    xr(SC,L)=xr(SC,L-1)+DRL

                    DL=L-SL
                    DRC=float(dis(2)-DL)*(R10-R00)+float(DL)*(R11-R01)
                    DRC=DRC/float(dis(1)*dis(2))
                    do 84 while (C.lt.(ps(1)+dis(1)*ix))
                        C=C+1
                        xr(C,L)=xr(C-1,L)+DRC
84                  continue
83              continue
```

```
81          continue
80      continue
c       ---------------------------------------------------------------
c       EXTRAPOLATION
c       ---------------------------------------------------------------
        do 85 L=1,ps(2)-1
            do 86 C=ps(1),pe(1)
                dl=ps(2)-L
c               xr(C,L)=xr(C,ps(2))+float(dl)/float(dis(2))
c    &                                *(xr(C,ps(2))-xr(C,ps(2)+dis(2)))
                xr(C,L)=xr(C,ps(2))
86          continue
85      continue
        do 90 L=pe(2)+1,iwy
            do 91 C=ps(1),pe(1)
                dl=L-pe(2)
c               xr(C,L)=xr(C,pe(2))+float(dl)/float(dis(2))
c    &                                *(xr(C,pe(2))-xr(C,pe(2)-dis(2)))
                xr(C,L)=xr(C,pe(2))
91          continue
90      continue

        do 92 L=1,iwy
            do 93 C=1,ps(1)-1
                dc=ps(1)-C
c               xr(C,L)=xr(ps(1),L)+float(dc)/float(dis(1))
c    &                                *(xr(ps(1),L)-xr(ps(1)+dis(1),L))
                xr(C,L)=xr(ps(1),L)
93          continue
            do 94 C=pe(1)+1,iwx
                dc=C-pe(1)
c               xr(C,L)=xr(pe(1),L)+float(dc)/float(dis(1))
c    &                                *(xr(pe(1),L)-xr(pe(1)-dis(1),L))
                xr(C,L)=xr(pe(1),L)
94          continue
92      continue
c       ---------------------------------------------------------------
c       ADD OFFSET
c       Since the offset value is subtracted from original shift value,
c       offset value is added to final shift value.
c       ---------------------------------------------------------------

        do L=1,iwy
            do C=1,iwx
                xr(C,L)=xr(C,L)+float(offset)
            end do
        end do

        return
        end
```

```
/*======================================================================
===
      Subroutine write_imageq
      Write Subtracted Image with various Format
      Coded by Shige
======================================================================
===*/
#include    <stdio.h>
#include        <string.h>
#include    <stdlib.h>
#include    <time.h>
#include        "TempSub.H"

void  prepare_FCRimage(short*, int*, int*, int, short*, int, int);
void    prepare_THVimage(short*, int, int);
void    prepare_NOHimage(short*, int, int);
void  modify_header(char*, char*, DisplayParameter);
void  modify_preDispPara(char*);
extern      NumTime     GetNumTime();
extern      StudyPara   ReadTSubDefFile(char*);
extern      FcrStand    getFcrStandInfo(FILE*, char*);
extern      FcrDicom    getFcrDicomInfo(FILE*, char*);
extern      ThvDicom    getThvDicomInfo(FILE*, char*);
extern      void        putThvDicomInfo(char*, ThvDicom);

#ifdef KRL_GNU
  #define write_imageq write_imageq__
#else
  #define write_imageq write_imageq_
#endif

extern "C" void write_imageq(short *image, int  *p_col, int *p_lin,
                        char  *DefFile,
                        char  *PreImage, char *CurImage,
                        char  *SubImage)
{
FILE    *fp;
short *FCRImage;
char  *PreFcrStdHeader = new char[HEADER_SIZE];
char  *CurFcrStdHeader = new char[HEADER_SIZE];
char  *PreFcrDcmHeader = new char[MAX_FCR_DCM_HEADER_SIZE];
char  *CurFcrDcmHeader = new char[MAX_FCR_DCM_HEADER_SIZE];
char  *PreThvDcmHeader = new char[MAX_THV_HEADER_SIZE];
char  *CurThvDcmHeader = new char[MAX_THV_HEADER_SIZE];

FcrStand    PreFcr, CurFcr;
FcrDicom    PreDcm, CurDcm;
ThvDicom    PreThv, CurThv;
StudyPara   studyPara = ReadTSubDefFile(DefFile);
NumTime           numTime = GetNumTime();

/*-------------------------------------------------------------------
      OUTPUT_STANDARD_FCR_HEADER/OUTPUT_DICOM_THV_HEADER
--------------------------------------------------------------------*/
if (studyPara.outputImageFormat == OUTPUT_STANDARD_FCR_HEADER) {
  printf("\a\aOUTPUT_STANDARD_FCR_HEADER is not permitted!!\n");
  exit(0);
   }
else if (studyPara.outputImageFormat == OUTPUT_DICOM_THV_HEADER) {
  prepare_THVimage(image, *p_col, *p_lin);
   }
```

```
else if (studyPara.outputImageFormat == OUTPUT_NON_HEADER) {
  prepare_NOHimage(image, *p_col, *p_lin);
   }

/*-------------------------------------------------------------------
      Read Header Information
--------------------------------------------------------------------*/
if   (studyPara.inputImageFormat == INPUT_STANDARD_FCR_HEADER) {
  if ((fp = fopen(CurImage,"r")) == NULL) {
    printf("%s is missing\n", CurImage); exit(0);
     }
  CurFcr = getFcrStandInfo(fp, CurFcrStdHeader);
  fclose(fp);
  if ((fp = fopen(PreImage,"r")) == NULL) {
    printf("%s is missing\n", PreImage); exit(0);
     }
  PreFcr = getFcrStandInfo(fp, PreFcrStdHeader);
  fclose(fp);
   }
else if (studyPara.inputImageFormat == INPUT_DICOM_FCR_HEADER) {
  if ((fp = fopen(CurImage,"r")) == NULL) {
    printf("%s is missing\n", CurImage); exit(0);
     }
  CurDcm = getFcrDicomInfo(fp, CurFcrDcmHeader);
  fclose(fp);
  if ((fp = fopen(PreImage,"r")) == NULL) {
    printf("%s is missing\n", PreImage); exit(0);
     }
  PreDcm = getFcrDicomInfo(fp, PreFcrDcmHeader);
  fclose(fp);
   }
else if (studyPara.inputImageFormat == INPUT_DICOM_THV_HEADER) {
  if ((fp = fopen(CurImage,"r")) == NULL) {
    printf("%s is missing\n", CurImage); exit(0);
     }
  CurThv = getThvDicomInfo(fp, CurThvDcmHeader);
  fclose(fp);
  if ((fp = fopen(PreImage,"r")) == NULL) {
    printf("%s is missing\n", PreImage); exit(0);
     }
  PreThv = getThvDicomInfo(fp, PreThvDcmHeader);
  fclose(fp);
   }

/*-------------------------------------------------------------------
      Determine Output FileName
--------------------------------------------------------------------*/
if      (studyPara.subFName == SUB_FNAME_FIX) {
  sprintf(SubImage, "%s%s", studyPara.subDirectory,
studyPara.subFixFName);
   }
else if (studyPara.subFName == SUB_FNAME_PATIENT_ID) {
  numTime.year = numTime.year - START_YEAR;
  sprintf(SubImage, "%s%s_%04d%02d%02d_%02d%02d%02d",
                studyPara.subDirectory, CurFcr.PatID,
                numTime.year, numTime.month, numTime.day,
                numTime.hour, numTime.minute, numTime.second);
   }
else if (studyPara.subFName == SUB_FNAME_PATIENT_NAME_LONG) {
  sprintf(SubImage, "%s%s.%s.%s.%s.%s.sub",
        studyPara.subDirectory, CurDcm.PatName, PreDcm.Date,
```

```
PreDcm.Time,
                                      CurDcm.Date, CurDcm.Time);
   }
else if (studyPara.subFName == SUB_FNAME_PATIENT_NAME_SHORT) {
  sprintf(SubImage, "%s%s.%s.%s.sub",
        studyPara.subDirectory, CurDcm.PatName, PreDcm.Date,
CurDcm.Date);
   }
else if (studyPara.subFName == SUB_FNAME_PATIENT_ID_DATE) {
  sprintf(SubImage, "%ss%s.%s.%s.sub",
        studyPara.subDirectory, CurThv.PatID, PreThv.Date, CurThv.Date);
   }

/*-------------------------------------------------------------------
      Modify Header Information
--------------------------------------------------------------------*/
if (studyPara.outputImageFormat == OUTPUT_STANDARD_FCR_HEADER) {
  modify_header(PreFcrStdHeader, CurFcrStdHeader,
studyPara.subDispPara);
   }
else if (studyPara.outputImageFormat == OUTPUT_DICOM_THV_HEADER) {
  sprintf(CurThv.Date, "%04d%02d%02d",
      numTime.year, numTime.month, numTime.day);
  sprintf(CurThv.Time, "%02d%02d%02d.000000",
      numTime.hour, numTime.minute, numTime.second);
  CurThv.Row  = *p_lin;
  CurThv.Col  = *p_col;
  CurThv.NBit = 10;
  strcpy(CurThv.Dir, PA_THV);
  putThvDicomInfo(CurThvDcmHeader, CurThv);
   }

/*-------------------------------------------------------------------
      Save Subtraction Image onto Disk
--------------------------------------------------------------------*/
printf("        SubImage:\t%s\n", SubImage);
if ((fp = fopen(SubImage,"w")) == NULL) {
  printf("%s can't be opened\n", SubImage); exit(0);
   }
if (studyPara.outputImageFormat == OUTPUT_STANDARD_FCR_HEADER) {
  fwrite(PreFcrStdHeader, sizeof(char),  HEADER_SIZE, fp);
  #ifdef  _SUN_SPARC_SOLARIS_1X_
    fwrite((char*)FCRImage, sizeof(short), FCR_WX * FCR_WY, fp);
  #else
    fwrite(FCRImage, sizeof(short), FCR_WX * FCR_WY, fp);
  #endif
  delete FCRImage;
   }
else if (studyPara.outputImageFormat == OUTPUT_DICOM_THV_HEADER) {
  fwrite(CurThvDcmHeader, sizeof(char), CurThv.HeaderSize, fp);
  #ifdef  _SUN_SPARC_SOLARIS_1X_
    fwrite((char*)image, sizeof(short), *p_col * *p_lin, fp);
  #else
    fwrite(image, sizeof(short), *p_col * *p_lin, fp);
  #endif
   }
else {
  #ifdef  _SUN_SPARC_SOLARIS_1X_
    fwrite((char*)image, sizeof(short), *p_col * *p_lin, fp);
  #else
    fwrite(image, sizeof(short), *p_col * *p_lin, fp);
```

```
  #endif
   }
fclose(fp);

/*-------------------------------------------------------------
      Change File Protection
--------------------------------------------------------------*/
system("rm -f temp.cmd");
if ((fp = fopen("temp.cmd", "w") ) == NULL) {
  printf("\7\7temp.dat can't be opened\n"); exit(0);
   }
fprintf(fp, "chmod 666 %s\n", SubImage);
fclose(fp);
system("chmod +x temp.cmd");
system("temp.cmd");
system("rm -f temp.cmd");

/*-------------------------------------------------------------
      Change Display Parameter of Previous Image
--------------------------------------------------------------*/
if (studyPara.inputImageFormat == INPUT_STANDARD_FCR_HEADER &&
    studyPara.orgImageTransfer == ON &&
    studyPara.preDispPara      == BLACK_OUT)
    modify_preDispPara(PreImage);

/*-------------------------------------------------------------
      DELETE
--------------------------------------------------------------*/
delete PreFcrStdHeader; delete CurFcrStdHeader;
delete PreFcrDcmHeader; delete CurFcrDcmHeader;
delete PreThvDcmHeader; delete CurThvDcmHeader;
}

/*================================================================
        Subroutine prepare_NOHimage
        Byte-swap pixels, if necessary.  I'm not clipping to MaxGray,
        because I don't think that the code did so before (and I'm
        trying to mimic old code behavior) -- Roger
=================================================================*/
void    prepare_NOHimage(short *image, int col, int lin)
{
  /* With this file format, pixels should be stored as high-endian.
Thus,
     if the machine is low endian, we need to byte swap. */
  int shouldByteSwap = _isMachineLowEndian;

  if (shouldByteSwap)
    {
    for (int i = 0; i < col * lin; i++)
      image[i] = (((image[i]) & 0xFF00) >> 8) |
                 (((image[i]) & 0x00FF) << 8);
    }

}

/*================================================================
      Subroutine prepare_FCRimage
      Expand, Flip and Inverse Gray Scale for FCR Format
=================================================================*/
void  prepare_FCRimage(short *image,    int *p_col, int *p_lin, int
ratio,
                 short *FCRImage, int mszx,   int mszy)
{
}

/*=================================================================
      Subroutine modify_header
      Modify Exam date in FCR Header and Determine Output FileName
====================================================================*/
void  modify_header(char *PreFcrStdHeader,
                  char *CurFcrStdHeader,
                  DisplayParameter subDispPara)
{
}

/*=================================================================
      Subroutine modify_preDispPara
      Modify Display Parameter of Previous image to BLACK_OUT
====================================================================*/
void  modify_preDispPara(char *PreFName)
{
}

/*============================================================
      Subroutine prepare_THVimage
      Byte Swap and Inversion Gray Scale
===============================================================*/
void prepare_THVimage(short *image, int col, int lin)
{
  /* With this file format, pixels should be stored as low-endian.
Thus,
     if the machine is high endian, we need to byte swap. */
  int shouldByteSwap = !_isMachineLowEndian;

int   i;
short   MaxGray = 1023;

for (i = 0; i < col * lin; i++)
  {
  if (shouldByteSwap)
    image[i] = (((image[i] ^ MaxGray) & 0xFF00) >> 8) |
               (((image[i] ^ MaxGray) & 0x00FF) << 8);
  else
    image[i] = image[i] ^ MaxGray;
  }
}
```

PRELIMINARY PROGRAM

SCAR 2003

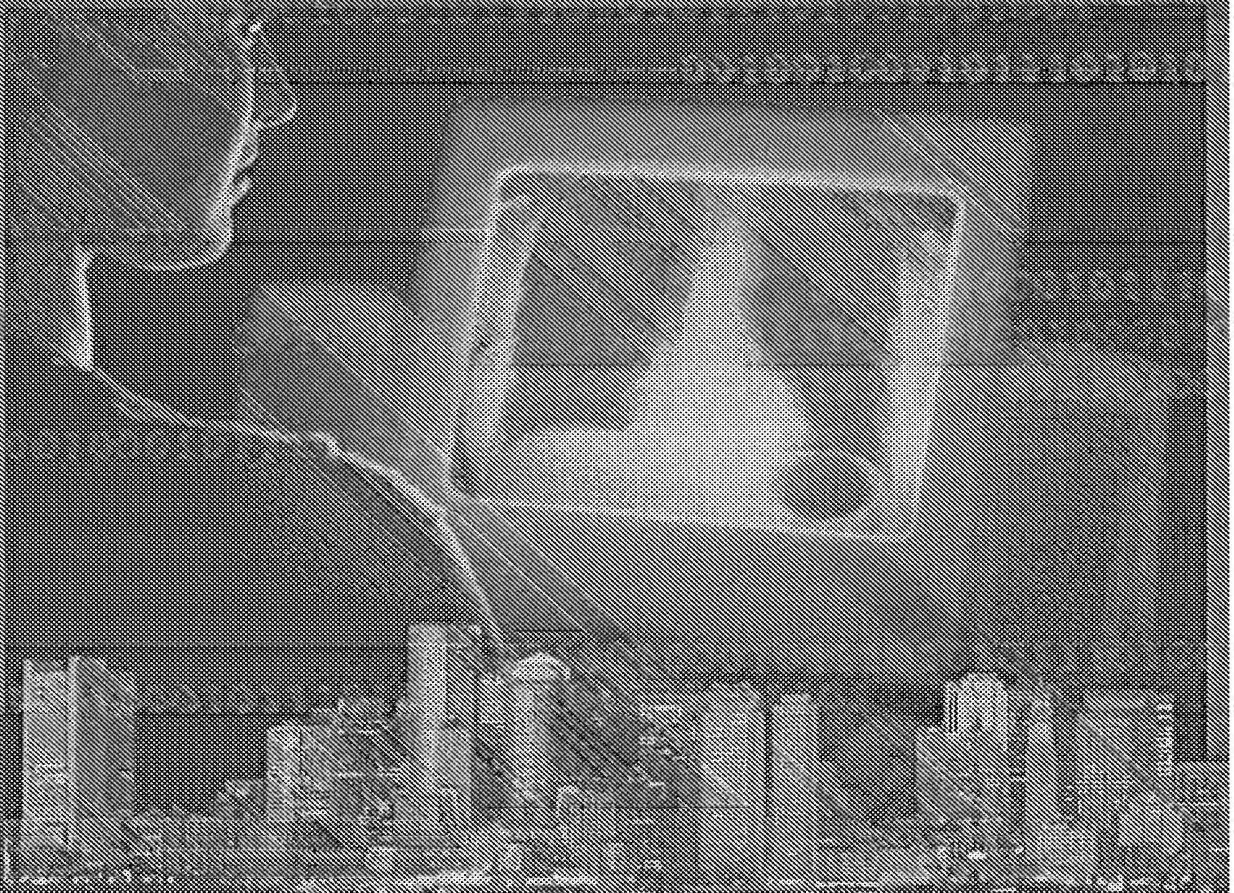
The 20th Symposium for Computer Applications in Radiology

ANNUAL MEETING 2003

JUNE 7–10, 2003 • BOSTON, MASSACHUSETTS

The Annual Meeting of the Society for Computer Applications in Radiology (SCAR)

Jointly Sponsored by the American College of Radiology (ACR) and SCAR



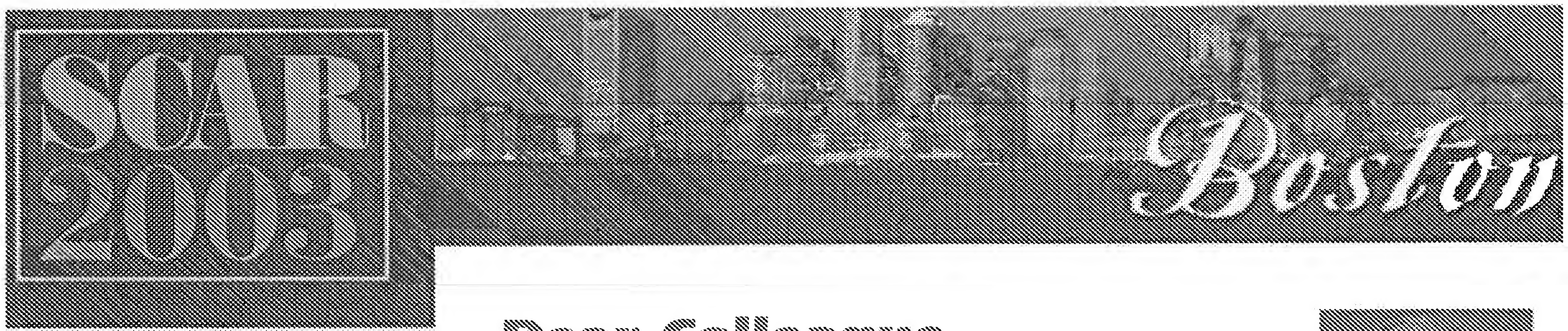
SCAR

Educating Healthcare Professionals for Tomorrow's Technology

Boston

2003 TOUR HOST INSTITUTIONS:

- Beth Israel Deaconess Hospital
- Brigham and Women's Hospital
- Children's Hospital of Boston
- Massachusetts General Hospital
- New England Baptist Hospital



SCAR Annual Meeting Program Committee

CHAIR
Byrn Williamson, Jr., MD
Mayo Clinic, Rochester

EDUCATIONAL CO-CHAIRS
Bruce I. Reiner, MD
University of Maryland
Eliot L. Siegel, MD
Baltimore VAMC, University of Maryland

SCIENTIFIC CO-CHAIRS
Bradley J. Erickson, MD, PhD
Mayo Clinic, Rochester
Steve G. Langer, PhD
Mayo Clinic, Rochester

TECHNICAL EXHIBITS CHAIR
Richard L. Morin, PhD
Mayo Clinic, Jacksonville

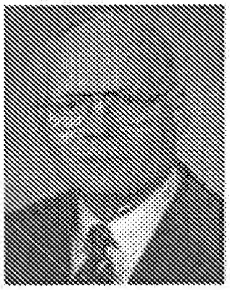
LIAISON – CME
John A. Carrino, MD, MPH
Brigham and Women's Hospital

Bradford J. Richmond, MD
The Cleveland Clinic Foundation

LOCAL COMMITTEE
Ramin Khorasani, MD
Brigham and Women's Hospital
David Hirschorn, MD
Massachusetts General Hospital
S. Ted Treves, MD
Children's Hospital Boston
David A. Quinlan, RT(R)
Beth Israel Deaconess Medical Center
Ronald D. Williams, RT(R)
New England Baptist Hospital

STAFF
Anna Marie Mason,
[illegible]

Dear Colleague,



Do you need access to authoritative, up-to-date information on PACS, electronic imaging, and radiology information systems? Would you like to view exhibits of leading radiology computer products and services in a relaxed atmosphere? If so, plan now to attend SCAR 2003, the 20th Symposium on Computer Applications in Radiology, which will be held in Boston, June 7–10, 2003. The Symposium is the longest running series of meetings on radiology computer applications. Since the first conference in 1964, SCAR has been the gathering place for experts in the field and those who want to learn from them. Attendees tell us that they enjoy the diversity of educational programs, the interactive and dynamic sessions, and the extensive technical exhibits. Despite the growth of the meeting in recent years, SCAR retains a collegial and relaxed atmosphere that attendees appreciate. This year, the Program Committee has continued to strengthen and expand the program. Highlights of SCAR 2003 include:

- Keynote address by Mr. Ray Kurzweil, pioneer of computer-based speech recognition systems, author of *The Age of Intelligent Machines* and *The Age of Spiritual Machines*, and a 2002 inductee into the National Inventors Hall of Fame.
- An updated SCAR University, a comprehensive didactic course in computer applications in radiology. Introductory, intermediate, and advanced lectures will cover the latest information on PACS, RIS, CAD, speech recognition and CR/DR. Many institutions bring their entire PACS team — administrators, CIO's, radiologists and technologists — to SCAR U for the latest information on technology implementation in radiology.
- Closing Session on the developing information explosion in radiology. How can radiologists deal with the burgeoning numbers of images produced by new technology? Representatives of NASA, the CIA, and the entertainment industry will explain how they manage huge data sets.
- More than 75 original scientific papers, posters, and demonstrations on all aspects of computer applications in radiology, from image processing to PACS implementation to speech recognition systems.
- Tours of electronic imaging activities at Beth Israel Deaconess Medical Center, Brigham and Women's Hospital, Children's Hospital of Boston, Massachusetts General Hospital, and New England Baptist Hospital. Special sessions delivered by radiology informatics faculty of Boston medical schools.
- A large exhibit hall filled with technical exhibits of leading radiology image management and IT vendors. More PACS vendors assembled than any other conference except the RSNA.

Whether you are a radiologist considering incorporating electronic systems into your practice, a CIO or administrator evaluating the costs and benefits of such systems, or a computer scientist involved in radiology research, SCAR 2003 will provide the up-to-date information you need. Please review the enclosed program and plan to attend this premiere meeting for users and developers of computer-based equipment and applications in medical imaging. CME credit is available for attendees. We look forward to seeing you in Boston!

BW illiamson

Byrn Williamson, Jr., MD
Chair, SCAR Program Committee

Course Objectives

The Symposium for Computer Applications in Radiology is an annual scientific and educational meeting presented by the Society for Computer Applications in Radiology (SCAR), and it is designed to provide important information to professionals who use, buy, or develop computer-based equipment with applications in radiology. It features the most recent developments in medical computer applications, particularly the advances in computer technology that improve the clinical practice of radiology and the effective management of health care resources.

Upon completion of the program, participants will be prepared to:

- Determine which computer applications can contribute to their practice
- Evaluate components of electronic image and information management systems
- Prepare for the changes that will result from implementing computer applications in their departments and institutions
- Choose promising areas for future research

Continuing Medical Education Information

Physician:
This activity has been planned and implemented in accordance with the Essentials and Standards of the Accreditation Council for Continuing Medical Education through joint sponsorship of the American College of Radiology and the Society for Computer Applications in Radiology. The American College of Radiology is accredited by the ACCME to provide continuing medical education for physicians.

The ACR designates this educational activity for up to 28 hours of Category 1 credit towards the AMA's Physician's Recognition Award. Each physician should claim only those hours of credit that he/she actually spends in the educational activity.

Technologists:
The American College of Radiology (ACR) is approved by the American Registry of Radiologic Technologists (ARRT) as a Recognized Continuing Education Evaluation Mechanism (RCEEM) to sponsor and/or review Continuing Medical Educational programs for Radiologic Technologists and Radiation Therapists.

The ACR designates this Continuing Medical Educational Activity as meeting the criteria for up to 28 Category A credit hours of the ARRT. Each technologist should claim only those hours of credit that he/she actually spends in the educational activity.

Medical Physicists:
The American College of Radiology is accredited by the Accreditation Council for Continuing Medical Education to sponsor continuing medical education credit activities.

The ACR designates the following educational activity as meeting the criteria for up to 28 hours of Medical Education for Physicists (MEPS) credit. Each medical physicist should claim only those hours of credit that he/she actually spends in the educational activity.

Presenters at this conference will disclose any conflict of interest or their intention to discuss off-label use, if applicable, in accordance with ACCME Standards and FDA requirements. Conflict of interest will be disclosed either in print or verbally at the beginning of the presentation.

Dates to Remember

Hotel Reservation Deadline	April 28
Early Registration Deadline	May 2
Cancellation/Refund Deadline	May 16
Pre-Registration Deadline	May 30
On-Site Registration Opens	June 6
PACS Administration Course	June 6
Meeting Dates	June 7–10

Table of Contents

Who Should Attend SCAR 2003

- Radiologists and other physicians who are considering implementing RIS, PACS, speech recognition, or teleradiology systems in their practice. (The "Program-at-a-Glance" indicates sessions of particular interest to practicing radiologists with an asterisk*)
- Radiologists, imaging physicists, and others who are interested in learning about cutting edge electronic imaging developments
- Technologists, PACS administrators, and those who are interested in becoming PACS administrators
- CEO's, CFO's, CIO's, and healthcare administrators at institutions that are considering implementing or replacing PACS or Radiology Information Systems
- Computer scientists, IT professionals, and engineers who want information about the latest research in computer applications in radiology
- *Anyone* who wants access to practical information about imaging technology in an open, collegial environment

General Sessions

OPENING SESSION

Saturday, June 7, 2003
8:00 AM – 9:30 AM
Grand Ballroom
Sheraton Boston Hotel

Welcome

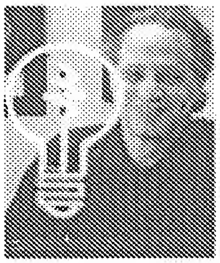
Katherine P. Andriole, PhD
Chair, Society for Computer Applications in Radiology
University of California, San Francisco

Byrn Williamson, Jr., MD
Chair, Annual Meeting Program Committee
Mayo Clinic, Rochester

Keynote Address

"The Impact of 21st Century Technology on Human Health and Society"

Raymond Kurzweil
Founder, Chairman and CEO
Kurzweil Technologies, Inc.



Raymond Kurzweil is an inventor of computer-based speech recognition technology and futurist author of the best selling book, *The Age of Spiritual Machines, When Computers Exceed Human Intelligence,* as well as a 2002 inductee into the National Inventors Hall of Fame.

In "The Impact of 21st Century Technology on Human Health and Society," Mr. Kurzweil will investigate the implications of the accelerating knowledge of technology and workings of the human brain. Once non-biological intelligence matches the range and subtlety of human intelligence, it will necessarily soar past it because of the continuing acceleration of information-based technologies, as well as the ability of machines to instantly share their knowledge.

Intelligent nanorobots will be deeply integrated in the environment, our bodies and our brains, providing vastly improved health, extended longevity, full-immersion virtual reality and enhanced human intelligence. The implication will be an intimate merger between the technology-creating species and the evolutionary process it spawned.

CLOSING SESSION

Tuesday, June 10, 2003
8:30 AM – 10:00 AM
10:30 AM – 12:30 PM
Constitution Ballroom, Sheraton Boston Hotel

Medical Image Interpretation — The Collision between Humans and Data

Co-Moderators:

Katherine P. Andriole, PhD
University of California, San Francisco
Chair, Society for Computer Applications in Radiology

Richard L. Morin, PhD
Mayo Clinic, Jacksonville
Chair, Transforming the Radiological Interpretation Process (TRIP) Subcommittee of the SCAR Research and Development Committee

Panel:

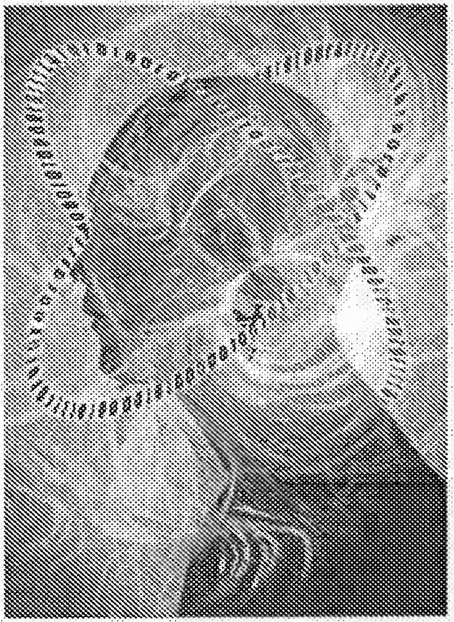
Richard Weinberg, PhD
University of Southern California

Gus Hunt
Central Intelligence Agency

Stephen Wharton, PhD
NASA Goddard Space Flight Center

William Young, MS
National Imagery & Mapping Agency

TBA
Disney Speaker via live satellite (tentative)



This is a session highlighting the SCAR 2003 theme of *Information Explosion: Embracing our Future* and will feature exploration of the management of large data sets by the intelligence, space exploration and entertainment industries.

Medical imaging data has increased radically in both the size of the examination as well as the number of examinations. This has resulted in display and analysis of increasingly greater amounts of image data by a radiologist each day. This situation currently appears unbounded and portends disaster for the future. The purpose of this session is to examine how other disciplines faced with similar challenges of large amounts of image data have dealt with these situations. SCAR intends for this session to spark strategic thinking and debate regarding a shift in the current paradigm used for medical image interpretation. SCAR is prepared to champion this cause and provide leadership to address and solve this dilemma.

Learning Objectives:

- Understand the problems associated with human viewing of large image data sets.
- Learn how other disciplines have solved problems associated with large image data sets.
- Engage in new methods for the interpretation of large image data examination in medical imaging.

Special Sessions

Presented by pioneering radiology informatics faculty of Boston medical schools

Saturday, June 7
Special Session I
10:00 AM – 11:45 AM

Strategic Business Plan for PACS

S. Ted Treves, MD
Session Chair
Children's Hospital of Boston
Chief, Division of Nuclear Medicine

This session will present the business perspective on assessing the value of PACS.

Sunday, June 8
Special Session II
10:00 AM – 11:45 AM

Decision Support Beyond Radiology

David W. Bates, MD, MSc
Session Chair
Medical Director of Clinical and Quality Analysis
Partners Healthcare System

Participants:

John Halamka, MD
Chief Information Officer
Caregroup Healthcare System

Gilad J. Kuperman, MD, PhD
Associate Director,
Clinical Informatics Research and Development
Partners Healthcare System

This session will describe clinical decision support in two Boston-area integrated delivery systems. Dr. Kuperman will begin by describing the basics of clinical decision support. Next, Dr. Halamka will describe Caregroup's efforts in this area. Dr. Bates will conclude describing evidence that clinical decision support makes a difference.

Learning Objectives:

- Describe different levels of clinical decision support, from simple to more complex.
- Describe evidence that clinical decision support improves the efficiency, quality and safety of care.

Monday, June 9
Special Session III
10:00 AM – 11:45 AM

Radiology Frontiers

Ramin Khorasani, MD
Session Chair
Vice Chairman, Department of Radiology
Medical Director, Multi disciplinary PACS
Assistant Professor of Radiology
Harvard Medical School
Brigham and Women's Hospital

Participants:

Thomas H. Lee, MD
Medical Director
Partners Community HealthCare, Inc.
Associate Editor, New England Journal of Medicine
Associate Professor of Medicine
Harvard Medical School

Steven E. Seltzer, MD
Chairman, Department of Radiology
Brigham and Women's Hospital
Phillip H. Cook Professor of Radiology
Harvard Medical School

Health care delivery systems are focusing on reducing medical errors and improving quality of care. Medical imaging will increasingly become a focus of these efforts due to rapid growth of imaging technologies and applications. Radiology has a unique opportunity to lead such efforts. We will discuss how leveraging information technology solutions can help transform radiology's perceived role of technology provider to that of a knowledge provider in patient care.

Learning Objectives:

- Recognize factors contributing to overuse/miss-use/under-use of imaging.
- Discuss medical errors in the context of medical imaging and their potential impact on health care costs and quality.
- Use a case example to describe how radiology can leverage information technology solutions, specifically Computerized Physician Order Entry, to help deliver knowledge at the point of care to reduce errors and improve quality of care while improving efficiency.

Monday, June 9
Special Session IV
1:15 PM – 3:00 PM

The Electronic Medical Record

Keith J. Dreyer, DO, PhD
Session Chair
Vice Chair,
Radiology Computing & Information Sciences
Massachusetts General Hospital
Partners HealthCare System, Inc.

Participants:

John P. Glaser, PhD
Vice President and Chief Information Officer
Partners HealthCare System, Inc.

John Halamka, MD
Chief Information Officer
Caregroup Healthcare System

Paul J. Chang, MD
Director, Division of Radiology Informatics
University of Pittsburgh Medical Center

Eliot L. Siegel, MD
Vice Chair
Imaging Information Systems
University of Maryland School of Medicine
Chief, Imaging VA Maryland Healthcare System

This session will summarize the current status and existing limitations in clinical practice for RIS/HIS integrated, enterprise-wide EMR.

Monday, June 9
Special Session V
3:30 PM – 5:15 PM

Radiology Systems Upgrades — the 7-Year Itch

David Avrin, MD, PhD
Session Chair
Professor of Radiology
Chief, Abdominal Imaging
University of Utah Hospitals & Clinics

The session will address the issues involved in renewing or even replacing major computer systems in radiology (i.e. PACS or RIS). A broad range of questions will be addressed: Which vendor upgrades do you accept? How do you finance a whole new system? How do you switch to a new system without bringing the radiology department to its knees?

Lunch Sessions

LUNCH SESSION I

Saturday, June 7
11:45 AM – 1:15 PM

Zen and the Art of PACS Administration

Moderator:

Paul G. Nagy, PhD
Medical College of Wisconsin

Panel:

Steve Chechet
Application Analyst
Appleton Medical Center/Thedacare

Charles Socia RT(R)(CT)(QM)
PACS System Administrator
Baptist Health

Marc Deshaies
Lead PACS Service Engineer
GE Medical Systems

Are you a PACS administrator who wants to glean secrets of success from PACS administrators who make it look easy? What is the career development path for PACS administrators? PACS is a constantly changing field, so if you do not stay up-to-date, you will become obsolete. This session will present several successful administrators who will discuss what you can do today to stay ahead of the curve.

Learning Objectives:

- Describe traits and prerequisites to be a PACS professional.
- Learn valuable job survival skills to succeed as a PACS professional.
- Explore tools and tips on how to become a PACS expert.

LUNCH SESSION II

Saturday, June 7
11:45 AM – 1:15 PM

Public Domain Software for PACS and Informatics Implementation

Moderator:

J. Anthony Seibert, PhD
University of California, Davis

Participants:

R. L. "Skip" Kennedy, MSc
Kaiser Permanente, Sacramento

Steven C. Horii, MD
University of Pennsylvania

Public domain software, including freeware, shareware and open source development opportunities is a resource that can provide significant assistance to implementers and users of PACS. This session will describe methods, strategies, and the discovery process of finding software and information from simple DICOM image viewers, database management tools, PACS administrative and QC tools, DICOM libraries, PACS FAQs, to available RFP documents and open source code. Open source code development for use and contribution by the public sector provides a fertile ground for creative implementation and sharing of ideas. Demonstration of specific software programs during the session will illustrate many of these creative and state-of-the-art capabilities.

Learning Objectives:

- Learn about the public domain software available from the Web and other resources.
- Gain an understanding of how to access, download, and implement specific freeware and shareware routines and retrieve useful information and suggestions about PACS and informatics issues.
- Demonstrate the capabilities, advantages, and implications of open source software.

LUNCH SESSION III

Sunday, June 8
11:45 AM – 1:15 PM

How Not to Give a Scientific Talk

Presented by the SCAR Research and Development Committee

Moderator:

Katherine P. Andriole, PhD
University of California, San Francisco
Chair, SCAR Research and Development Committee

Participants:

John A. Carrino, MD, MPH
Brigham and Women's Hospital

Bradley J. Erickson, MD, PhD
Mayo Clinic, Rochester

Bruce I. Reiner, MD
University of Maryland

This session will illustrate the proper way to orally present a scientific paper. Examples of good and poor presentations of the same paper will be given. Ample time will be available to analyze each presentation, pointing out the positive and negative aspects with suggestions for improvement. The audience will be invited to participate in the discussion.

Learning Objectives:

- Learn the basic elements required to construct a solid scientific oral paper presentation.
- Become aware of common mistakes and potential pitfalls to avoid in giving scientific presentations.
- Recognize good and bad practices in scientific presentations to facilitate improving the creation and delivery of presentations in the future.

LUNCH SESSION IV

Sunday, June 8
11:45 AM – 1:15 PM

IHE Update—Integrating the Healthcare Enterprise: Workflow and How You Get It

An RSNA IHE Initiative Presentation

Moderator:

David Piraino, MD
Cleveland Clinic Foundation

Panel:

David Channin, MD
Northwestern Memorial Hospital

Paul Nagy, PhD
Froedert Hospital

Kevin O'Donnell
Toshiba Medical Systems

John Paganini
IDX Systems Corporation

Charles Parisot
GE Medical Systems

IHE is an initiative of IT and healthcare professionals and industry to implement standards to solve real-world clinical problems. In radiology IHE has defined standards-based transactions to support the workflow of typical patient encounters and numerous other enhancements to the efficiency of clinical care. IHE makes it simpler for vendors to adopt these solutions and for purchasers to specify them when acquiring systems—reducing the difficulty and cost of tightly integrating systems. Learn what you need to know to put these benefits into practice today.

Learning Objectives:

- Learn the benefits of a tightly integrated workflow in radiology.
- Understand the constantly expanding scope of integration capabilities available through IHE.
- Draw on the experience of users who have successfully integrated imaging and information systems with IHE.

LUNCH SESSION

Monday, June 9
11:45 AM – 12:15 PM

SCAR Membership Meeting and Fellows Induction

12:15 PM – 1:15 PM

3rd SCAR Research and Development Committee Symposium

"Evaluation of Interstitial Lung Disease on 5 Mpixel CRT vs. 3 Mpixel LCD Displays"

Presented by the SCAR Research and Development Committee

Moderators:

Katherine P. Andriole, PhD
University of California, San Francisco
Chair, SCAR Research and Development Committee

Steve G. Langer, PhD
Mayo Clinic, Rochester

Participants:

Bradley J. Erickson, MD, PhD
Brian J. Bartholmai, MD
Ken A. Fetterly, MS
Mayo Clinic, Rochester

Eliot L. Siegel, MD
University of Maryland School of Medicine

John A. Carrino, MD
Brigham and Women's Hospital

The R&D study is aimed at performing a receiver operator characteristic (ROC) evaluation of 5 Mpixel CRT vs. the evolving 3 Mpixel LCD display. The question to be answered is, "Can radiologists perform diagnosis as accurately on the currently available 3 Mpixel LCD technology as they do with current 5 Mpixel CRT displays?" Initial results from a multi-institution, multi-observer study will be presented.

Learning Objective:

- Learn the basics of statistical concepts, experimental design, and informed display choices.

BREAKFAST SESSION

2003 RESIDENT ROUNDTABLE

Tuesday, June 10
7:00 AM – 8:30 AM

Electronic Teaching Tools: Old Dogs and New Tricks

Presented by the SCAR Resident and Fellows Education and Training Committee

Moderator:

David S. Channin, MD
Northwestern University Medical School
Chair, SCAR Resident Education and Training Committee

Participants:

Brian J. Bartholmai, MD
Mayo Clinic, Rochester

Barton F. Branstetter, MD
University of Pittsburgh Medical Center

David S. Hirschorn, MD
Massachusetts General Hospital

Khan M. Siddiqui, MD
Geisinger Medical Center

Participants will have an opportunity to hear a summary of existing electronic teaching tools in radiology. There will be a chance to discuss individual experiences with these tools. A focus will also be made on functional requirements for new tools to be developed in the future.

Learning Objectives:

- Understand the electronic teaching tools available now.
- Describe what is missing from teaching tools.
- Understand technologies available to develop new tools.

SCAR U
Educating Healthcare Professionals for Tomorrow's Technology

SCAR University 2003

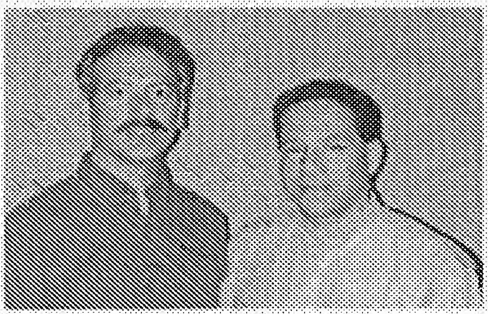
The advancement of computer applications in medicine continues to move at a breathtaking rate, and no discipline is more affected (or more at the cutting edge) than radiology.

SCAR has been fortunate to have some of the leading researchers and adopters of these technologies within its constituency, and these experts have served as educators within SCAR University. The clinical and research topics of special interest to the members of SCAR continue to expand into new arenas such as computer-aided diagnosis, structured reporting, speech recognition, and strategies for review and interpretation of large and complex image datasets.

With the increasing adoption of digital radiography and PACS, filmless imaging is moving from an early adopter to an early majority phase. What was previously the exclusive domain of tertiary care academic facilities is now entering into the domain of small and medium sized community hospitals. As the electronic medical record also becomes a reality, these digital applications will become necessary components to achieve this paperless, filmless paradigm.

At the same time, our society has expanded its base to include healthcare professionals in a wide array of occupations including physicists, technologists, administrators, engineers, physicians, information technology specialists, and industry consultants. We welcome this expanded member base, which serves to facilitate the sharing of ideas across the entire healthcare spectrum.

In response to the rapid developments in the field, SCAR University continues to expand its curricula to meet these rapidly expanding educational challenges. We have added a number of new educational tracks to our program, while expanding the educational program to go above and beyond the annual meeting. Some of these new educational initiatives can be found on-line at www.scarnet.org, as well as in print with our primer series, which includes four publications to date (Security, Electronic Archive, Quality Assurance, and our newest addition on the topic of Electronic Reporting).


Eliot Siegel, MD and Bruce Reiner, MD
Co-Chairs, SCAR University

We would like to thank you for being an active member of SCAR University and hope you enjoy the meeting. We continue to look for new ways to meet your educational needs and welcome any suggestions or feedback you have to offer.

100 LEVEL INTRODUCTORY COURSES

Will provide attendees with the basics and fundamental information used in everyday application of the technologies.

SATURDAY, JUNE 7

10:00 AM – 11:45 AM

101
The Essentials of CR & DR
Katherine Andriole, PhD
University of California, San Francisco
10:00 AM – 10:30 AM

102
Use of Decision Support Tools in Today's Clinical Practice
Curtis Langlotz, MD, PhD
University of Pennsylvania
10:30 AM – 11:00 AM

103
CR/DR Workflow Optimization
Anna Chacko, MD
The Lahey Clinic
11:00 AM – 11:30 AM

3:15 PM – 5:00 PM

104
The LCD vs. CRT Conundrum
Michael Flynn, PhD
Henry Ford Health System
3:15 PM – 3:45 PM

105
Fundamentals of Teleradiology Quality Control
John Romlein, MS
Xtria Healthcare Systems
3:45 PM – 4:15 PM

106
Buyer's Guide to RIS Purchasing
William Montgomery, CIO
Shands HealthCare, Inc.
4:15 PM – 4:45 PM

SUNDAY, JUNE 8

10:00 AM – 11:45 AM

107
Introduction to Networking
Paul Chang, MD
University of Pittsburgh Medical Center
10:00 AM – 10:30 AM

108
Is Digital Mammography Ready for Prime Time?
TBA
10:30 AM - 11:00 AM

109
Introduction to Speech Recognition
Stephen Herman, MD
Toronto General Hospital
11:00 AM – 11:30 AM

1:15 PM – 3:00 PM

110
Digital Image Capture Using PACS
Richard Wiggins, III, MD
University of Utah School of Medicine
1:15 PM – 1:45 PM

111
Introduction to Storage: Does Size Really Matter?
Edward Smith, DSc
University of Rochester Medical Center
1:45 PM – 2:15 PM

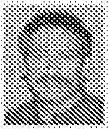
112
Designing and Redesigning the Digital Radiology Reading Room
Eliot Siegel, MD
Baltimore VAMC/University of Maryland
2:15 PM – 2:45 PM

SENIOR-LEVEL SESSIONS (200, 300, 400)

These senior sessions are organized by topic and will allow participants to explore digital imaging technologies in greater depth. These detailed and complex didactic offerings are aimed at the technophiles and more experienced users of the technology. Courses 201, 301, 401 through 212, 312, 412 parallel topics 101–112 in the 100 level introductory course session. 213, 313, 413 deal with an additional topic, security.

SATURDAY, JUNE 7

Design Considerations in a Filmless Enterprise



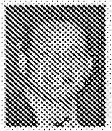
Eliot L. Siegel, MD
University of Maryland
Baltimore VAMC
Section 12 Head
10:00 AM – 11:45 AM

212
The Digital Imaging Department: An Architect's Perspective
Morris Stein, FAIA, FACHA
The Stein-Cox Group Architects
10:00 AM – 10:30 AM

312
Radiology Department Redesign in the Digital Era: A Case Study Approach
Bill Rostenberg, FAIA, FACHA
Smith Group Architects
10:30 AM – 11:00 AM

412
Looking into the Crystal Ball: The Radiology Department of the Not Too Distant Future
Mark Morita
GE Medical Systems
11:00 AM – 11:30 AM

Productivity/Workflow



Bruce I. Reiner, MD
University of Maryland
Section 3 Head
3:15 PM – 5:00 PM

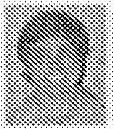
203
Interpretation Strategies for Large Imaging Datasets
Eliot Siegel, MD
University of Maryland
VA Maryland Healthcare System
3:15 PM – 3:45 PM

303
The New Paradigm in Electronic Reporting
Bruce Reiner, MD
University of Maryland
3:45 PM – 4:15 PM

403
Designing Software Tools for Radiologist Workflow Optimization
Kaushal Shastri
Fujifilm Medical Systems, USA
4:15 PM – 4:45 PM

SUNDAY, JUNE 8

New Frontiers in Digital Radiography



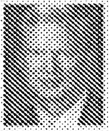
Katherine P. Andriole, PhD
University of California, San Francisco
Section 1 Head
10:00 AM – 11:45 AM

201
Purchasing and Implementation Strategies for Digital Radiography
R.L. "Skip" Kennedy, MSc
Kaiser Permanente Sacramento
10:00 AM – 10:30 AM

301
Specialty Applications
Katherine Andriole, PhD
University of California, San Francisco
10:30 AM – 11:00 AM

401
New Technologies in Digital Radiography
J. Anthony Seibert, PhD
University of California, Davis
11:00 AM – 11:30 AM

Radiologist Decision Support Tools



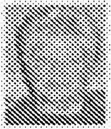
Bradley J. Erickson, MD, PhD
Mayo Clinic, Rochester
Section 2 Head
1:15 PM – 3:00 PM

202
Clinical Applications of CAD
Heber MacMahon, MD
University of Chicago
1:15 PM – 1:45 PM

302
Use of Advanced Image Processing Algorithms for Image Enhancement
Bradley Erickson, MD, PhD
Mayo Clinic, Rochester
1:45 PM – 2:15 PM

402
Neural Networks and Fuzzy Logic
Susan Wood, PhD
Jimmy Roehrig, PhD
R2 Technology, Inc.
2:15 PM – 2:45 PM

Information Systems



Janice Honeyman-Buck, PhD
University of Florida
Section 6 Head
3:30 PM – 5:15 PM

206
The Changing Role of Informatics in the Current Digital Radiology Practice
Chris Sistrom, MD
University of Florida
3:30 PM – 4:00 PM

306
Integration of RIS and PACS
Meryll Frost
Medical Imaging Consultants, Inc.
4:00 PM – 4:30 PM

406
Advanced Information System Functionality, Interoperability, and Issues
Janice Honeyman-Buck, PhD
University of Florida
4:30 PM – 5:00 PM

Security

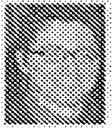
Samuel J. Dwyer, III, PhD
University of Virginia Health System
Section 13 Head
3:30 PM – 5:15 PM

213
HIPAA Security Update
Kristin Hughes, JD
SC&A Consulting, Inc.
3:30 PM – 4:00 PM

313
Security Strategies for Wireless Technologies
Samuel Dwyer, III, PhD
University of Virginia Health System
4:00 PM – 4:30 PM

413
Creating a Bullet-Proof Digital Imaging Network
Herman Oosterwijk, MS, MBA
OTech Inc.
4:30 PM – 5:00 PM

Workstation Design and Quality Control



John A. Carrino, MD, MPH
Brigham and Women's Hospital
Section 4 Head
3:30 PM – 5:15 PM

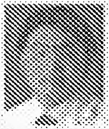
204
Assessment of Display Performance for Medical Imaging Systems
Andrew Maidment, PhD
University of Pennsylvania
3:30 PM – 4:00 PM

304
Developing an Enterprise-Wide Monitor QC Program
Manuel Arreola, PhD
University of Florida
4:00 PM – 4:30 PM

404
Comparison of Color and Monochrome Displays in 2003
Michael Flynn, PhD
Henry Ford Health System
4:30 PM – 5:00 PM

MONDAY, JUNE 9

Connectivity/Networking



Paul J. Chang, MD
University of Pittsburgh Medical Center
Section 7 Head
10:00 AM – 11:45 AM

207
Update on IHE
David Channin, MD
Northwestern University Medical School
10:00 AM – 10:30 AM

307
New DICOM Initiatives
Steven Horii, MD
University of Pennsylvania
10:30 AM – 11:00 AM

407
Wireless Technologies: Current State-of-the-Art
Paul Chang, MD
University of Pittsburgh Medical Center
11:00 AM – 11:30 AM

QA in the Digital Enterprise

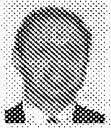
Charles E. Willis, PhD
Texas Children's Hospital
Section 5 Head
10:00 AM – 11:45 AM

205
Expanding the Role of the Technologist in Digital Radiography QC
Ellen Charkot, MRT
Hospital for Sick Children, Toronto
10:00 AM – 10:30 AM

305
Artifacts and Misadventures in Digital Radiography
Charles Willis, PhD
Texas Children's Hospital
10:30 AM – 11:00 AM

405
Developing an Enterprise-wide Digital Quality Assurance Program
Stephen Thompson, MS
MD Anderson Cancer Center
11:00 AM – 11:30 AM

Speech Recognition and Structured Reporting



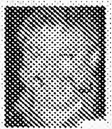
David L. Weiss, MD
Geisinger Medical Center
Section 9 Head
1:15 PM – 3:00 PM

209
Demo of Problem-Solving Scenarios
David Weiss, MD
Geisinger Medical Center
1:15 PM – 1:45 PM

309
Practical Applications of Structured Reporting with Demo
Curtis Langlotz, MD, PhD
University of Pennsylvania
1:45 PM – 2:15 PM

409
Radiology Lexicon and the RadLex Project
John Carrino, MD, MPH
Brigham and Women's Hospital
2:15 PM – 2:45 PM

Digital Mammography



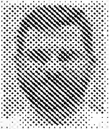
Martin J. Yaffe, PhD
Sunnybrook & Women's College Health Science Center
Section 8 Head
1:15 PM – 3:00 PM

208
Current and Future Technologies for Digital Mammography
Martin Yaffe, PhD
Sunnybrook & Women's College Health Science Center
1:15 PM – 1:45 PM

308
DICOM and PACS for Digital Mammography
Andrew Maidment, PhD
University of Pennsylvania
1:45 PM – 2:15 PM

408
Advanced Clinical Applications for Digital Mammography (Telemammography, Tomosynthesis, CAD Breast Angiography)
Daniel Kopans, MD
Massachusetts General Hospital
2:15 PM – 2:45 PM

SCAR U: How To (A Practical User's Guide)



Nogah Haramati, MD
Montefiore Medical Center
Section 10 Head
3:30 PM – 5:15 PM

210
An Update on Wireless Technologies
Mary McKenna, RN, MSN
Bellevue Hospital & South Manhattan Healthcare Network
3:30 PM – 4:00 PM

310
Interpretation Strategies for Large Cross Sectional Image Data Sets
Nogah Haramati, MD
Montefiore Medical Center

Menashe Benjamin, PhD
Algotec Systems, Inc.
4:00 PM – 4:30 PM

410
Customizing Hanging Protocols
Roberta Locko, MD
Harlem Hospital Center
4:30 PM – 5:00 PM

Electronic Storage Media

David S. Channin, MD
Northwestern University Medical School
Section 11 Head
3:30 PM – 5:15 PM

211
Storage Media
Katherine Andriole, PhD
University of California, San Francisco
3:30 PM – 4:00 PM

311
Storage Options: DAS, HSM, SAN, NAS and other Buzzwords
Prakash Mathew, PhD
GE Medical Systems
4:00 PM – 4:30 PM

411
Practical and Clinical Determinants of Storage Requirements
David Channin, MD
Northwestern University Medical School
4:30 PM – 5:00 PM

Pre-Conference, Educational Sessions

SCAR PACS Administration Course — NEW!

Friday, June 6, 2003 • Hynes Convention Center • 9:00 am–5:00 pm

Presented by the PACS Administration Subcommittee of the SCAR Education Committee

Faculty:

George Bowers, MBA
Principal
Health Care Information Consultants, LLC

Paul G. Nagy, PhD
Director, Radiology Informatics Laboratory
Medical College of Wisconsin

Jay Gaeta
PACS Consultant

Thomas M. Hanson, MS, RT
PACS Specialist
Froedtert Hospital

Liaisons:

Bruce I. Reiner, MD
Associate Professor, Department of Radiology
University of Maryland

Eliot L. Siegel, MD
Director Imaging, Department of Radiology
Baltimore VAMC/University of Maryland

Target Audience: Recent and soon-to-be PACS Administrators

Registration: $100 USD (meals and registration materials included). Space is limited. Register now and reserve your space by using the SCAR 2003 Meeting Registration Form on pages 25–26 of this brochure or online on the SCAR Website. Six hours of continuing education credits available for course attendees.

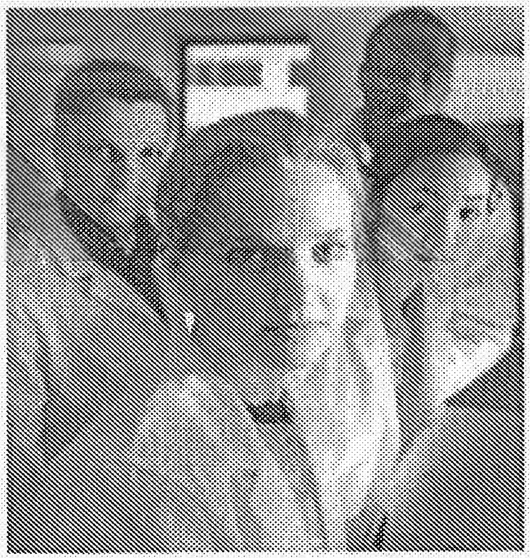
Course Description:

As the leading educational organization that deals with PACS, it is synergistic with SCAR's mission to define PACS Administration and determine the competencies required in this evolving field.

The first SCAR PACS Administration Course will be a focused one-day session prior to the SCAR 2003 Annual Meeting. The course will provide an overview of the profession covering the four modules that encompass the different competencies. The course is designed to provide a framework that enables the participant to develop an understanding of the roles and skill sets necessary for effective PACS Administration. Tailored specifically for new PACS Administrators, this introductory course will identify resources available, including a roadmap to the SCAR Annual Meeting particularly focused on PACS Administration content.

Since there is way too much content to teach people PACS administration in a single day, the SCAR PACS Administration course will present an overview of the methods and tools a person can use to be a competent and happy PACS administrator. This course will focus on the techniques to embrace change and stay on top of the fast moving field of PACS Administration.

For more information, visit the SCAR Website.



Learning Objectives:

By the conclusion of the course, participants will be able to:

- Describe the core competencies of PACS Administration.
- Recognize each of the roles involved in PACS Administration: Users, Business, and Technical.
- Identify the Available Resources, including SCAR 2003 conference sessions of particular interest to PACS Administrators.

SCAR-affiliated User Groups — Annual Educational Meetings

IDXrad Radiology Information Systems Society (IRISS)

Thursday, June 5 – Friday, June 6, 2003
Sheraton Boston Hotel, 9:00 am–5:00 pm

AGFA PACS Users Group (APUG)

Friday, June 6, 2003
Hynes Convention Center, 9:00 am–5:00 pm

FUJI PACS User Group Meeting

Friday, June 6, 2003
Hynes Convention Center, 9:00 am–5:00 pm

Program at a Glance

Day 1—Saturday, June 7

6:30 am	Registration 6:30 am–5:30 pm	Continental Breakfast 7:00 am–8:00 am		
8:00 am	**Opening Session** The Impact of 21st Century Technology on Human Health and Society *Keynote: Ray Kurzweil* **8:00 am–9:30 am** • Grand Ballroom Sheraton Boston Hotel			
9:30 am	Break 9:30 am–10:00 am			
10:00 am	* **SCAR U 100** Introductory Course **10:00 am–11:45 am** **101** The Essentials of CR & DR **102** Use of Decision Support Tools in Today's Clinical Practice **103** CR/DR Workflow Optimization	**SCAR U Senior Session** Design Considerations in a Filmless Enterprise **10:00 am–11:45 am** **212** The Digital Imaging Department: An Architect's Perspective **312** Radiology Department Redesign in the Digital Era: A Case Study Approach **412** Looking into the Crystal Ball: The Radiology Department of the Not Too Distant Future	**Special Session I** Strategic Business Plan for PACS *S. Ted Treves, MD* **10:00 am–11:45 am**	
11:45 am	**Lunch Session 1** Zen and the Art of PACS Administration **11:45 am–1:15 pm**	**Lunch Session 2** Public Domain Software **11:45 am–1:15 pm**	**Lunch** **11:45 am–1:15 pm**	
1:15 pm	**Scientific Session 1** Image Processing **1:15 pm–2:45 pm**	* **Scientific Session 2** Departmental Productivity & Workflow **1:15 pm–2:45 pm**	**Scientific Session 3** Image Acquisition & Storage **1:15 pm–2:45 pm**	* **Hospital Tours A** **Tour 1,2,3,4** **1:15 pm–3:15 pm**
2:45 pm	Break 2:45 pm–3:15 pm			
3:15 pm	* **SCAR U 100** Introductory Course **3:15 pm–5:00 pm** **104** The LCD vs. CRT Conundrum **105** Fundamentals of Teleradiology Quality Control **106** Buyer's Guide to RIS Purchasing	**SCAR U Senior Session** Productivity & Workflow **3:15 pm–5:00 pm** **203** Interpretation Strategies for Large Imaging Datasets **303** The New Paradigm in Electronic Reporting **403** Designing Software Tools for Radiologist Workflow Optimization		* **Hospital Tours B** **Tour 1,2,3,4** **3:15 pm–5:15 pm**
5:00 pm	**SCAR 2003 Opening Reception in the Exhibit Hall** 5:00 pm–7:00 pm			

* Activity of particular interest to practicing radiologists.

Day 2—Sunday, June 8

7:00 am	Registration 7:00 am–5:00 pm	Continental Breakfast 7:00 am–8:00 am		
8:00 am	* **Scientific Session 4** Reading Room **8:00 am–9:30 am**	**Scientific Session 5** Enterprise Productivity & Workflow **8:00 am–9:30 am**	**Scientific Session 6** Image Distribution **8:00 am–9:30 am**	
9:30 am	Break 9:30 am–10:00 am	Exhibit Hall A		
10:00 am	* **SCAR U 100** Introductory Course **10:00 am–11:45 am** **107** Introduction to Networking **108** Is Digital Mammography Ready for Prime Time? **109** Introduction to Speech Recognition	**SCAR U Senior Session** New Frontiers in Digital Radiography **10:00 am–11:45 am** **201** Purchasing and Implementation Strategies for Digital Radiography **301** Specialty Applications **401** New Technologies in Digital Radiography	**Special Session II** Decision Support Beyond Radiology *David Bates, MD, MSc* **10:00 am–11:45 am**	
11:45 am	**Lunch Session 3** How Not to Give a Scientific Talk **11:45 am–1:15 pm**	**Lunch Session 4** IHE Update **11:45 am–1:15 pm**	**Lunch** **11:45 am–1:15 pm** Exhibit Hall A	
1:15 pm	* **SCAR U 100** Introductory Course **1:15 pm–3:00 pm** **110** Digital Image Capture Using PACS **111** Introduction to Storage: Does Size Really Matter **112** Designing and Redesigning the Digital Radiology Reading Room	**SCAR U Senior Session** Radiologist Decision Support Tools **1:15 pm–3:00 pm** **202** Clinical Applications of CAD **302** Use of Advanced Image Processing Algorithms for Image Enhancement **402** Neural Networks and Fuzzy Logic	**Poster and Demo Session** **1:15 pm–3:00 pm**	* **Hospital Tours C** **Tour 1,2,3,4** **1:15 pm–3:15 pm**
3:00 pm	Break 3:00 pm–3:30 pm			
3:30 pm	**SCAR U Senior Session** Information Systems **3:30 pm–5:15 pm** **206** The Changing Role of Informatics in the Current Digital Radiology Practice **306** Integration of RIS and PACS **406** Advanced Information System Functionality, Interoperability, and Issues	* **SCAR U Senior Session** Security **3:30 pm–5:15 pm** **213** HIPAA Securing Update **313** Security Strategies for Wireless Technologies **413** Creating a Bullet-proof Digital Imaging Network	**SCAR U Senior Session** Workstation Design and Quality Control **3:30 pm–5:15 pm** **204** Assessment of Display Performance for Medical Imaging Systems **304** Developing an Enterprise-wide Monitor QC Program **404** Comparison of Color and Monochrome Displays in 2003	**Hospital Tours D** **Tour 1,2,3,4** **3:15 pm–5:15 pm**
6:00 pm	**SCAR Welcome Reception 6:00 pm–8:00 pm**	**Prudential Center Top of the Hub**		

* Activity of particular interest to practicing radiologists.

Program at a Glance

Day 3—Monday, June 9

7:00 am	Registration 7:00 am–5:00 pm	Continental Breakfast 7:00 am–8:00 am		
8:00 am	**Scientific Session 7** Vendor Session **8:00 am–9:30 am**	* **Scientific Session 8** PACS Experience **8:00 am–9:30 am**	**Scientific Session 9** Infrastructure & Administration **8:00 am–9:30 am**	
9:30 am	Break 9:30 am–10:00 am			
10:00 am	**SCAR U Senior Session** Connectivity/Networking **10:00 am–11:45 am** **207** Introduction to Networking **307** New DICOM Initiatives **407** Wireless Technologies: Current State-of-the-Art	**SCAR U Senior Session** Quality Assurance in the Digital Enterprise **10:00 am–11:45 am** **205** Expanding the Role of the Technologist in the Digital Radiology QC **305** Artifacts and Misadventures in Digital Radiography **405** Developing an Enterprise-wide Digital Quality Assurance Program	* **Special Session III** Radiology Frontiers *Ramin Khorasani, MD* **10:00 am–11:45 am**	
11:45 am	**SCAR Membership Meeting and Fellows Induction** 3rd SCAR Research and Development Committee Symposium **11:45 am–1:15pm**		**Lunch** **11:45 am–1:15 pm** Exhibit Hall A	
1:15 pm	**SCAR U Senior Session** Speech Recognition and Structured Reporting * **1:15 pm–3:00 pm** **209** Introduction to Speech Recognition **309** Demo of Problem-solving Scenarios **409** Radiology Lexicon and the RadLex Project	**SCAR U Senior Session** Digital Mammography **1:15 pm–3:00 pm** * **208** Current and Future Technologies for Digital Mammography **308** DICOM and PACS for Digital Mammography **408** Advanced Clinical Applications for Digital Mammography	**Special Session IV** The Electronic Medical Record (EMR) *Keith Dreyer, DO, PhD* **1:15 pm–3:00 pm**	* **Hospital Tours E** **Tour 5** **1:15 pm–3:15 pm**
3:00 pm	Break 3:00 pm–3:30 pm	Exhibit Hall A		
3:30 pm	**SCAR U Senior Session** * SCAR U: How To (A Practical User's Guide) **3:30 pm–5:15 pm** **210** An Update on Wireless Technologies **310** Interpretation Strategies for Large Cross Sectional Image Data Sets **410** Customizing Hanging Protocols	**SCAR U Senior Session** Electronic Storage Media **3:30 pm–5:15 pm** **211** Storage Media **311** Storage Options: DAS, HSM, SAN, NAS and other Buzzwords **411** Practical and Clinical Determinants of Storage Requirements	* **Special Session V** System Upgrades the 7-Year Itch *David Avrin, MD, PhD* **3:30 pm–5:15 pm**	* **Hospital Tours F** **Tour 5** **3:15 pm–5:15 pm**

* Activity of particular interest to practicing radiologists.

Day 4—Tuesday, June 10

7:00 am	Registration 7:00 am–9:00 am	Continental Breakfast 7:00 am–8:00 am	**Residents' Roundtable** Electronic Teaching Tools: Old Dogs and New Tricks **7:00 am–8:30 am**
8:30 am	* **Closing Session** Medical Image Interpretation — The Collision between Humans and Data *Co-Moderators: Richard Morin, PhD, Katherine Andriole, PhD* **8:30 am–10:00 am**		
10:00 am	Break 10:00 am–10:30 am		
10:30 am	* **Closing Session** Medical Image Interpretation — The Collision between Humans and Data *Co-Moderators: Richard Morin, PhD, Katherine Andriole, PhD* **10:30 am–12:30 pm**		

* Activity of particular interest to practicing radiologists.

Key:

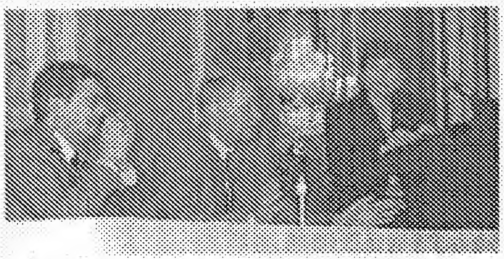
□ General and Special Sessions

▤ SCAR University Sessions

▦ Scientific Sessions

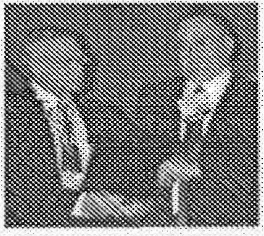
▩ Hospital Tours

Please note: the program sessions are preliminary and subject to change or substitutions.



Dr. Alan Rowberg, Dr. Brad Erickson, and Dr. Kathy Andriole field R&D Symposium attendee questions.

SCAR Annual Membership Meeting



Dr. David Avrin of the University of California, San Francisco (left) presents a plaque to Samuel Dwyer, PhD (right) of the University of Virginia, honoring him as a SCAR fellow.

SCAR Members — be sure to attend the Annual Membership Meeting/Luncheon and induction of SCAR Fellows on Monday, June 9 at 11:45 AM. Following the business meeting, the SCAR Research and Development Committee will present their third symposium. The topic of the 2003 R&D Symposium is *"Evaluation of Interstitial Lung Disease on 5 Mpixel CRT vs. 3 Mpixel LCD."*

Registration Hours:

Friday, June 6	**5:00 PM – 8:00 PM**
Saturday, June 7	**6:30 AM – 5:30 PM**
Sunday, June 8	**7:00 AM – 5:00 PM**
Monday, June 9	**7:00 AM – 5:00 PM**
Tuesday, June 10	**7:00 AM – 9:00 AM**

All scientific and educational sessions, posters and demonstrations for SCAR 2003 will be held at the Sheraton Boston Hotel. Technical exhibits will be held in Hall A at the Hynes Convention Center. Bus transportation will be provided from the Boylston Street entrance of the Hynes Convention Center to the hospital tour sites.

For a complete up-to-date list of presentations
and online registration, visit **www.scarnet.org**

SCAR gratefully acknowledges the following cooperating organizations:

AAPM	American Association of Physicists in Medicine
ACR	American College of Radiology
EuroPACS	
HIMSS	Healthcare Information and Management Systems Society
RSNA	Radiological Society of North America
SPIE	The International Society of Optical Engineering

Stay informed!

For the latest information on SCAR 2003 or to register online:

- Visit the SCAR Website at **www.scarnet.org**
- Call the SCAR office at **703-757-0054**; Monday – Friday, 9:00 AM – 6:00 PM eastern time
- Email SCAR at **SCAR2003@scarnet.org**

Drs. Erickson, Rowberg, Horii, Andriole and Siegel participate in a special session of 'experts' at SCAR 2002.

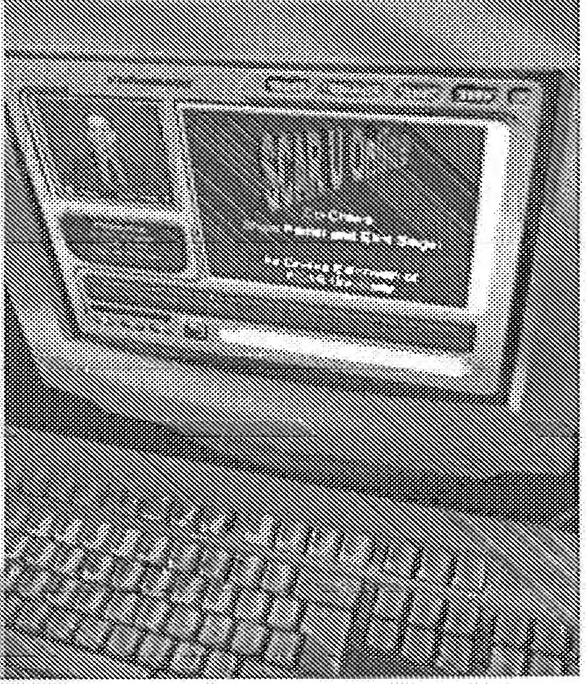
Get a Head Start on SCAR 2003

at SCAR University Online...

Learn Virtually Anytime, Anywhere!

Fulfill your continuing education goals by taking courses online using SCAR's new learning program, SCAR U Online. Learn what you want, when you want, wherever you have computer access. Preview a course to see how easy online learning can be!

SCAR
Educating Healthcare Professionals for Tomorrow's Technology



SCAR Members save on registration fees...
only $75 for a full year of SCAR U introductory courses online

Scientific Papers

SATURDAY, JUNE 7

Session 1
Image Processing
1:15 PM – 2:45 PM

Co-Chairs:

Katherine P. Andriole, PhD
Bradley J. Erickson, MD, PhD

Utility of Advanced Computed Radiography Image Processing Algorithms in the Soft-Copy Interpretation of Musculoskeletal Trauma

Bruce I. Reiner, MD
University of Maryland
Eliot Siegel, MD
Ryan Moffitt
Steven Brower

Mining of Association Rules in Medical Image Data Sets

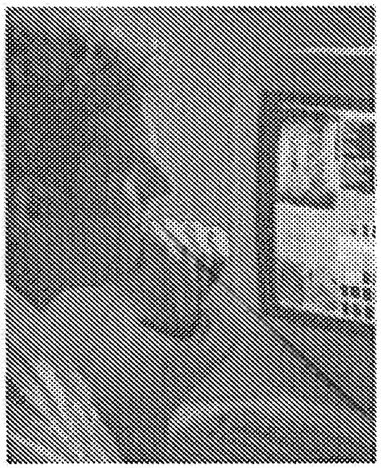
Sylvanus A. Ehikioya, PhD
University of Manitoba
Adepele Olukunle

A Distributed Execution Environment for Analysis of DCE-MR Image Datasets

Tahsin M. Kurc, PhD
Ohio State University
John D Fleig, MSc
Joel H Saltz, MD, PhD
Michael Knopp, MD, PhD

Detection of Microcalcification in Digitized Mammograms Using Wavelet Transform Local Extrema

M. G. Mini, MS
Cochin University of Science and Technology
V.P. Devassia
Tessamma Thomas, PhD



Session 2
Departmental Productivity & Workflow*
1:15 PM – 2:45 PM

Co-Chairs:

Bruce I. Reiner, MD
Charles E. Willis, PhD

Vendor Requirements for Implementation of the IHE Presentation of Grouped Procedures Integration Profile in a Multi-Vendor Environment

Gary J. Wendt, MD, MBA
University of Wisconsin, Madison
Wally Peppler, PhD

Creating a Playbook for IHE Scheduled Workflow Operations

Arnon Makori, MD
Feinberg School of Medicine
Northwestern University
David S. Channin, MD

Impact of Electronic Signature of Radiology Reports on Timeliness of Final Report Availability

Luigi Lepanto, MD
Centre Hospitalier de l'Universite de Montreal
Pierre Robillard, MD
Jacques Lesage, MD

How to Successfully Implement Voice Recognition: A Case Study at Children's Hospital Boston

Sharon E. Antiles, MPH
Children's Hospital Boston
Chuck Hornberger, MBA
Farhad Shahrooz
Robert Bramson, MD

Changes in Radiology Resident On-call Workflow After Implementation of PACS

Khan M. Siddiqui, MD
Geisinger Medical Center
Rodney G. Shaffer, MD
Faaiza Mahmoud, MD

Multi-Center Evaluation of Technologist Productivity and Workflow in Filmless Operation: A Comparison of Computed and Direct Radiography

Bruce I. Reiner, MD
University of Maryland
Eliot Siegel, MD
Frank Hooper, PhD

Session 3
Image Acquisition & Storage
1:15 PM – 2:45 PM

Co-Chairs:

Steve G. Langer, PhD
John A. Carrino, MD, MPH

Validation of a Self-Calibrating Active-Matrix Liquid Crystal Display System

Stephen L. Thompson, MS
MD Anderson Cancer Center
Charles E. Willis, PhD
Raimund Polman
Kenneth L. Homann

Mobile Screening Mammography: What Size Detector is Needed?

Gary J. Whitman, MD
MD Anderson Cancer Center
Donna Moxley, MS
Dorothy Page, RT (R)
Jessica Foust

Impact of Repeat Analysis in PACS

Maria Elissa Elevado Blado
Texas Children's Hospital
Yinlin Ma
Rebecca Ann Corwin, RT(R)
Stephanie G. Carr

Security Middle-Ware Infrastructure for DICOM Images in Health Information Systems

Vijay N.V. Kallepalli
University of Manitoba
Sylvanus A. Ehikioya, PhD
Sergio Camorlinga, MSc
Jose Rueda, PhD

Analyzing Audit Logs — A Multidimensional Approach

Robert M. Coleman
Maine Medical Center
Matthew D. Ralston, MD
Alexander Szafran, MS

*Sessions of particular interest to practicing radiologists

Session 4
Reading Room*
8:00 AM – 9:30 AM

Co-Chairs:

Eliot L. Siegel, MD
Jihong Wang, PhD

Designing the Reading Room in an Academic Environment

Thomas M. Hanson, MS, RT
Froedtert Hospital
Paul G. Nagy, PhD
Laura Kreiner
Jeff Rehm

Effect of Illuminance at Eye Level on Monitor Black Level Luminance and Monitor Calibration

Kish Chakrabarti, PhD
CDRH/FDA
Richard V. Kaczmarek, MS
Jerry A. Thomas, MS

Are Consumer Grade Flat Panel Monitors Comparable to Medical Grade CRT Monitors for Primary Diagnosis of Abdominopelvic CT Exams?

David S. Hirschorn, MD
Massachusetts General Hospital
Keith J. Dreyer, DO, PhD
Thomas Schultz

High Volume Teleradiology Service: Focus on Radiologist Satisfaction

Elizabeth A. Krupinski, PhD
University of Arizona
Kevin McNeill, PhD
Kai Haber, MD
Theron Ovitt, MD

A Cost Effective Web-Based Teaching File System

Blair T. Henderson, MD
University of Manitoba

Defining a Digital Teaching File Workflow: Specifications for Software Development

Barton F. Branstetter, MD
University of Pittsburgh
David M. Lionetti
Paul J. Chang, MD

Session 5
Enterprise Productivity & Workflow
8:00 AM – 9:30 AM

Co-Chairs:

Richard L. Morin, PhD
Curtis P. Langlotz, MD, PhD

Leveraging the Intranet for an Imaging Department: Centralizing Information, Improving Communications and Operations, and Providing Access to Learning Resources

William Tellier
Children's Hospital Boston
Linda Poznauskis
Keith Strauss
Robert MacDougall

Improving Emergency and Radiology Interdepartmental Communications Through Clinical Information Systems Integration and the Application of Mobile Computing Technologies

Wyatt M. Tellis
University of California, San Francisco
Katherine P. Andriole, PhD
David E. Avrin, MD, PhD

Asynchronous Collaboration: An Enabling Technique for Improved Radiology Workflow

Barton F. Branstetter, MD
University of Pittsburgh
David M. Lionetti
Brian Paterson
Paul J. Chang, MD

Clinician Assessment of Productivity Changes Following Enterprise PACS Implementation in a Community Hospital

Kevin R. Kirsch, RT(R)(CT)
Poudre Valley Hospital
Jonathan Brown, RT
J. Raymond Geis, MD
Shelly A. Plowman

Progress Towards Paperless Radiology in the PACS Environment

Matthew D. Ralston, MD
Maine Medical Center
Robert Coleman

Web-based Outpatient Radiology Order Entry

Daniel I. Rosenthal, MD
Massachusetts General Hospital
Thomas J. Schultz
David S. Hirschorn, MD
Keith J. Dreyer, DO, PhD

Session 6
Image Distribution
8:00 AM – 9:30 AM

Co-Chairs:

Paul J. Chang, MD
Gary J. Wendt, MD, MBA

Enhancement of Enterprise Diagnostic Review with Integration into Electronic Medical Record

Kevin W. McEnery, MD
MD Anderson Cancer Center
Charles T. Suitor, MS
Stephen K. Thompson, MS
Stan Hildebrand

Challenges and Limitations of Clinical Image Distribution in an Enterprise Wide PACS Environment — A Two-Year Evaluation of Multiple Approaches at the University of Wisconsin

Gary J. Wendt, MD, MBA
University of Wisconsin-Madison
Wally Peppler, PhD

Enterprise-wide Image Distribution: the BWH Experience — Ten Years and Counting

William Hanlon, MSc
Brigham and Women's Hospital
Ramin Khorasani, MD
Stephanie Hoogasian

Tools for Managing Image Flow in the Modality to Clinical-Image-Review Chain

Kenneth W. Clark, MS
Washington University
David L. Melson, MS
Stephen M. Moore, MS
G. James Blaine, DSc

Implementation of Key Image Note in PACS — Potential Problems and Solutions

Gary J. Wendt, MD, MBA
University of Wisconsin-Madison
Wally Peppler, PhD

Measures of the Utility of a Clinical PACS: Comparison of Self-Reported Measures and Direct Measures of PACS Usage by Clinicians

Eric P. Tamm, MD
MD Anderson Cancer Center
Kevin McEnery, MD

*Sessions of particular interest to practicing radiologists

Session 7
Vendor Session
8:00 AM – 9:30 AM

Co-Chairs:
Samuel J. Dwyer, III, PhD
Ramin Khorasani, MD

Implementation of a SANS Architecture Within A PACS Environment
Roy Seabolt
AMICAS, Inc.
Darlene Long
Bruce Hall

PACS Direct Experiences: Implementation, Selection and Benefits Realized
Karen Ondo
KLAS Enterprises
Ralph Reyes

PACS With HIS/RIS Integration in Community Hospitals
David W. Parker
SmartPACS
Kenneth C. Cohen, MD
Ann Hooper, RT
Steve Walter, MBA

Developing a Teaching File Authoring System Using Content Management Technology
Rex Jakobovits, PhD
Workhest Data Solutions
Mark Halsted, MD
Mark Shanaman, MS
Edward Weinberger, MD

PC-Based Ultrasound Image Acquisition and Data Archiving System Using Integrated Microelectronics
Steven R. Broadstone, DSc
Terason Division of Teratech Corporation
Xingbai He, PhD
Peter P. Chang, PhD
Alice M. Chiang, PhD

Implementing PACS: The Importance of Project Management: Tales from the Trenches
Stephen M. Doerner, RT
Kodak Health Imaging

Session 8
PACS Experience*
8:00 AM – 9:30 AM

Co-Chairs:
Paul G. Nagy, PhD
David S. Channin, MD

The PACS Pre-Implementation Process at a Major Teaching Hospital: A Multi-disciplinary Approach
D. Ben'et Gaytos
Children's Hospital Boston
John Speziale
Sharon Antiles, MPH
Robert Bramson, MD

Transitioning to a New PACS: 50 Ways to Leave Your Vendor
Barton F. Branstetter, MD
University of Pittsburgh
Claudine L. Martin
Therese A. Martin
Paul J. Chang, MD

Northwestern Year 4: Architectural Changes
Maria Z. Hernandez, RT(MR)
Northwestern Memorial Hospital
Elizabeth McKnight, RT (R)
Aimee Duvall, RT (R)
Andrew Longoria, RT (R)

Clinical Comparison of CRT and LCD Monitors in the Evaluation of Non-displaced Fractures
Bruce I. Reiner, MD
University of Maryland
Eliot Siegel, MD
Steven Brower
Ryan Moffitt

Can a PACS Workstation Work from 6,000 Miles Away?
David S. Hirschorn, MD
Massachusetts General Hospital
Charles D Levine, MD
Stephen R Baker, MD

PACS Modules Training at TCH
Maria Elissa Elevado Blado
Texas Children's Hospital
Stephanie G. Carr

Session 9
Infrastructure & Administration
8:00 AM – 9:30 AM

Co-Chairs:
Stephen K. Thompson, MS
Keith J. Dreyer, DO, PhD

Defining the PACS Profession
Paul G. Nagy, PhD
Medical College of Wisconsin
George Bowers
Bruce Reiner, MD
Eliot Siegel, MD

Negotiating a Service Level Agreement for PACS with the Enterprise
Bryant Mascarenhas, MBA
Froedtert Hospital
Paul G. Nagy, PhD
Daniel Peterson
Jeff Rehm

A Measurement Study of Diagnostic Imaging Modalities and Workgroups to Design a Suitable Enterprise PACS Network
Mpho Otukile
University of Manitoba-St. Boniface General Hospital Research Centre
Sergio Camorlinga, MSc
Jose Rueda, PhD

Paperless and Filmless: Integrating Dictation with PACS
Thomas E. Warfel, MD, PhD
University of Pittsburgh
Paul J. Chang, MD

Digital Image Conferencing in a Clinical Research Environment
Hendrik von Tengg-Kobligk, MD
Ohio State University
Klaus Baudendistel
William Bennett
D. Spigos

A Performance Study of Replicated Metadata for Implementing a Distributed PACS Patient Location System
Ellen Cheung
University of Manitoba-St. Boniface General Hospital Research Centre
Sergio Camorlinga
Ken Barker, PhD
Jose Rueda, PhD

Note: The program sessions are preliminary and are subject to change or substitutions
* Sessions of particular interest to practicing radiologists

Scientific Posters and Demonstrations

Poster and Demonstration Session:
Sunday, June 8, 2003
1:15 PM – 3:00 PM

Poster awards will be presented at the Sunday evening Welcome Reception.

A Fast Algorithm for the Cortical Surface Parameterization using Minimum Distance Field

Junki Lee, MS
Hanyang University
Jun-Soo Kwon, MD
Jong-Min Lee, PhD
Inyoung Kim, MD, PhD

A Novel Automatic Algorithm for Selecting a Standard Brain in a Data Set Using Simple Structure Analysis in Talairach Coordinate System

Jong-Min Lee, PhD
Hanyang University
Bangbon Koo
Sangmin Lee, PhD
Inyoung Kim, MD, PhD

A Perceptual Evaluation of JPEG2000 Image Compression for Digital Mammography

Sankararaman Suryanarayanan, MS
Emory University Medical School
Andrew Karellas, PhD
Srinivasan Vedantham, PhD
Hetal Ved

Automatic Detection and Segmentation of Low Contrast Objects in the Complex Background

Tatyana Belikova, PhD
Russian Academy of Sciences
Iryna Ivasenko, PhD
Roman Palenichka, PhD

Automatic Stitching of Digital X-ray Images

Alexander L. Berestov, PhD
Medical Canon Development
Srinivasan Gopasalamy, PhD
Ivan J. Bojer
Timothy L. Kohler

Boundary Segmentation for Detection of Spiculated Masses Using Morphological Characteristics in Digital Mammogram

Hosung Kim
Hanyang University
Jaehun Kim
Eunju Kim
Inyoung Kim, MD, PhD

ClubPACS: An Online Community for the PACS Administrator

Paul G. Nagy, PhD
Medical College of Wisconsin
Jeff Rehm
Charles E Kahn, MD

Comparison of Mammographic Imaging Systems in Detection of Simulated Microcalcifications: Flat Panel, CCD, and Screen/Film Combination

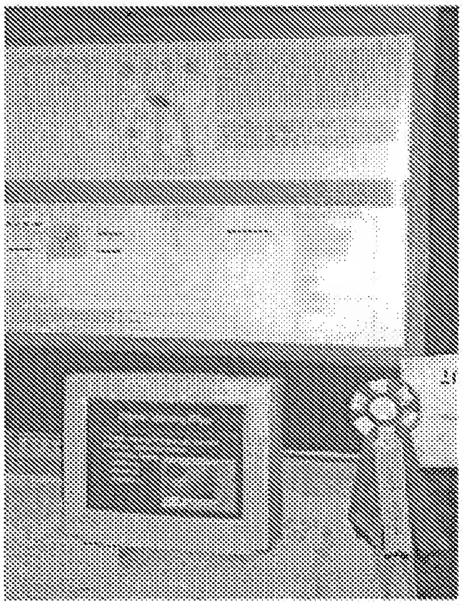
Gary J. Whitman, MD
MD Anderson Cancer Center
Chao-Jen Lai, PhD
Wei Tse Yang, MD
Elsa Arribas, MD

Development of ECG Management System Conformable to DICOM Waveform using XML

Yongho Cho, MSE
Hanyang University
Myoung-ju Jeon, MS
Hyungsik Choi, MD
Inyoung Kim, MD, PhD

Enterprise Imaging at Intermountain Health Care

Joe B. Boyce, MD
McKay-Dee Hospital, IHC
Deanna Welch
Mary Gathers
Darin Day



Evaluation of Automated and Semi-Automated Skull-stripping Algorithms: Similarity Index and Segmentation Error

Jong-Min Lee, PhD
Hanyang University
Jung-Hyun Kim
Ui-Cheul Yoon, MS
Inyoung Kim, MD, PhD

Modeling of Workflow in Diagnostic Radiology Departments

Spencer B. Gay, MD
University of Virginia Health System
Matthew J. Bassignani, MD
Alfred C. Weaver, PhD
C. Douglas Phillips, MD

Modern Technology Gives Birth to a New Nuclear Medical Imaging System Conception

Bouraoui Mahmoud, PhD
Faculty of Sciences of Monastir Tunisia
Med Hedi Bedoui
Radoslav Raychev
Habib Essabbah

Peak Signal to Noise Ratio - Performance Comparison of JPEG and JPEG 2000 for Various Medical Image Modalities, and Analysis of Precise Rate Distortion Capabilities for Improved Workflow Development

George P. Mulopulos, MD, FACR
Desert Radiologists
Laszlo R Gasztonyi
Albert Hernandez, MS

Planning for the Development of Telesonography

Matthew J. Bassignani, MD
University of Virginia Health System
Samuel J. Dwyer, PhD
Alfred C. Weaver, PhD
Jonathon Ciambotti, MD

Radiology Scheduling: Preferences of Users of Radiological Services and Impact on Referral Base Retention and Extension

Biswita C. Mozumdar, MD, MPH
National Institutes of Health
Douglas N. Hornsby, MD
Lisa Intriere, MD
Pablo Ros, MD, MPH

Softcopy Display Quality Comparison: A Proposed Quality-Index Curve

Jihong Wang, PhD
University of Texas Southwestern Medical Center
Qi Peng, MS

Teleradiology Use During Operation Joint Forge in Bosnia

Lance R. Williams, MD
Womack Army Medical Center

Transforming a Film-Based CT Practice to PACS: A CT PACS Trial Experience at Mayo Clinic Rochester

Suzanne K. Ramthun, MBA, RT(R)
Mayo Clinic, Rochester
Colleen M. Braun, RT(R)
Brian J. Bartholmai, MD

Transition from Film to Electronic Media in the First-Year Medical School Gross Anatomy Lab

Randy D. Ernst, MD
University of Texas Medical Branch at Galveston
Paul S. Sarat, MD
Orhan S. Ozkan, MD
Alberto Hernandez, MD

Use of Wireless PDA in Day-to-Day Radiology Practice

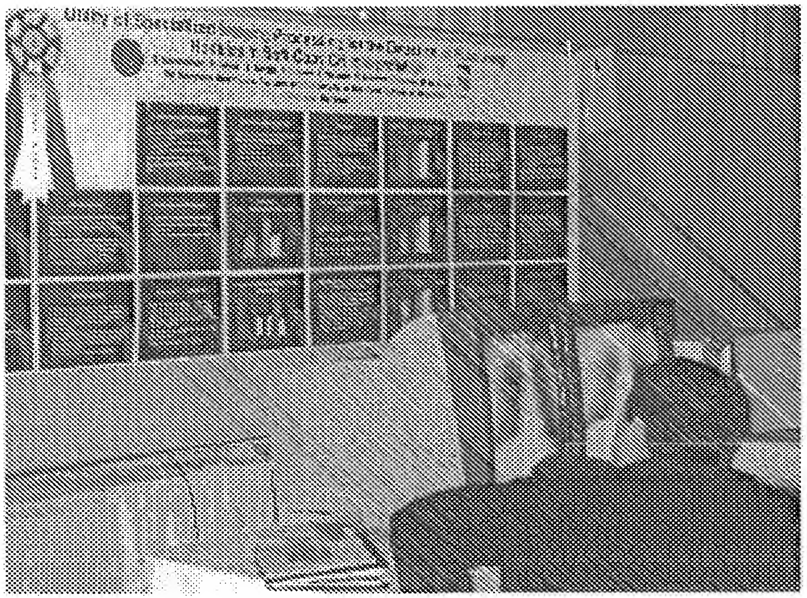
Khan M. Siddiqui, MD
Geisinger Medical Center
Joseph A. Scopelliti
Fredrick K. Emge, MD

Using Off the Shelf Digital Cameras to Scan Film into a Lightbox Free Environment

Yaron Rado, MD
Universitaetsklinik Duesseldorf
Benjamin Fritz, MD, DDS
Jens Nawatny, MD
Alexandra Rado, MD

Visualization of Three-Dimensional Fusion Image Using VRML in Clinical Epilepsy Case

Sang-Ho Lee
Research Institute for Radiological Sciences, Yonsei University
Dong-Hyun Kim
Sun Kook Yoo, PhD
Haijo Jung, PhD



For a complete up-to-date list of presentations, visit **www.scarnet.org**

Note: The poster and demo session is preliminary and subject to change or substitutions

SCAR 2003 Hospital Tours

Hospital Tours Schedule

Attendees of SCAR 2003 will have the opportunity to register for tours of Beth Israel Deaconess Medical Center (BID), Brigham and Women's Hospital (BWH), Children's Hospital of Boston (CHB), Massachusetts General Hospital (MGH) and New England Baptist Hospital (NEB).

Each institution has different vendors and configurations giving participants an overview of several system solutions. Tours will be offered as follows:

Tour 1 – BID

Tour 2 – BWH

Saturday, June 7, 2003
1:15 PM – 3:15 PM
3:15 PM – 5:15 PM

Tour 3 – CHB

Tour 4 – NEB

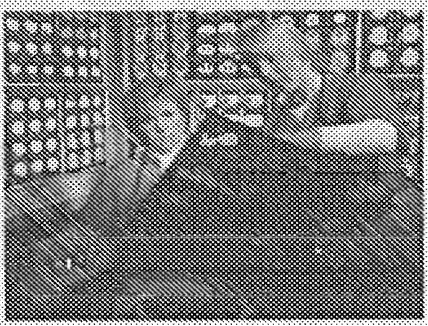
Sunday, June 8, 2003
1:15 PM – 3:15 PM
3:15 PM – 5:15 PM

Tour 5 – MGH

Monday, June 9, 2003
1:15 PM – 3:15 PM
3:15 PM – 5:15 PM

Bus transportation will be provided from Hynes Convention Center (Boylston Street entrance) to each of the medical centers. A tour of any one facility will last approximately one hour. Tours require advance registration and your tour times will be sent to you with your registration confirmation. Please make your tour selections on the SCAR 2003 Meeting Registration Form (page 26). There is no additional fee, but space is limited and tours will be assigned on a first-come first-serve basis. You may pre-register for a maximum of two tours.

If taking two tours, it is recommended that tours not be scheduled back-to-back, since participants must ride the bus back to the Hynes Convention Center before boarding another bus for the next tour. However, if a participant plans to visit both Brigham and Women's and Children's Hospital, it is recommended that they register for back-to-back tours. Due to the close proximity of BWH and CHB, tour guides will walk participants from one institution to the other.



Tour 1

BETH ISRAEL DEACONESS MEDICAL CENTER

Beth Israel Deaconess Medical Center is a major teaching affiliate of Harvard Medical School (since 1928). The center is a non-profit healthcare institute with 529 beds, 1200 physicians on the active medical staff and is renown for excellence in patient care, biomedical research, teaching and community service. Located in the heart of Boston's medical community it serves more than half a million patients annually in Boston and in communities North, West and South of the city.

The Division of Radiology at Beth Israel Deaconess Medical Center offers complete diagnostic services including general radiology, CT scans, MRI, ultrasound, mammography, nuclear medicine and interventional radiology. Each year over 250,000 examinations are performed and interpreted by sub-specialized radiologists.

The Division of Radiology has a PACS system that services three campuses and multiple outpatient centers. The PACS system has over 60 diagnostic workstations, 20 clinical workstations and Web-based image distribution. The Division of Radiology is electronically archiving 178,000 exams per year.

This tour will focus on digital workflow in an outpatient setting, providing participants with an opportunity to interact in a fully functional digital environment.

The tour includes:

Outpatient Workflow

- See CR and DR functioning in a busy outpatient department from patient arrival to final interpretation including remote reading with integrated RIS and softcopy interpretation.

Digital Mammography Workflow

- See how digital mammography improves workflow for the technologist and radiologist by streamlining the radiological process from the first patient contact through delivery of results.

Advanced Post Processing Methods

- See how a variety of advanced imaging techniques can be used to help with surgical planning, tumor staging and other facets of healthcare and research; also, the many ways in which image presentation can be made with multimedia.

The Digital Fileroom

- See how the fileroom duties have transformed from a film-based environment to digital.

BIDMC Vendors:

- GE Medical Systems
- FujiFilm Medical Systems
- Fischer Imaging Corporation
- PacsCube (DatCard Systems, Inc)
- Vital Images, Inc.

Tour 2

BRIGHAM AND WOMEN'S HOSPITAL

Since 1980, Brigham and Women's Hospital (BWH) has been recognized internationally for its excellence in patient care, medical research and the training of outstanding young physicians and other health care professionals. A teaching affiliate of Harvard Medical School and a founding member of Partners HealthCare System, Inc. (1994), the hospital comprises 716 beds, extensive outpatient facilities and state-of-the-art research laboratories.

The Department of Radiology offers a full spectrum of imaging services provided by sub-specialized radiologists. Each year over 500,000 examinations are performed. The Multidisciplinary PACS system at BWH services three hospitals (BWH, Dana-Farber Cancer Institute and Faulkner Hospital) with over 110 image acquisition devices and over 125 diagnostic and clinical review workstations, and Web-based image distribution. In addition to radiological images, OB-Gyn studies, cardiac catheterizations, echocardiography and vascular ultrasound studies are also archived to PACS. Further integration into the enterprise is being carried out via a physician order entry system and the electronic round trip.

Brigham and Women's Hospital would like to demonstrate the paperless/filmless workflow that is being implemented system-wide.

The tour includes:

Physician's Order

- Web Based Physician Order Entry used to deliver real-time decision support to referring physicians while enabling them to create clean and concise orders. Three exams will be scheduled directly on-line with radiology.

Imaging Services

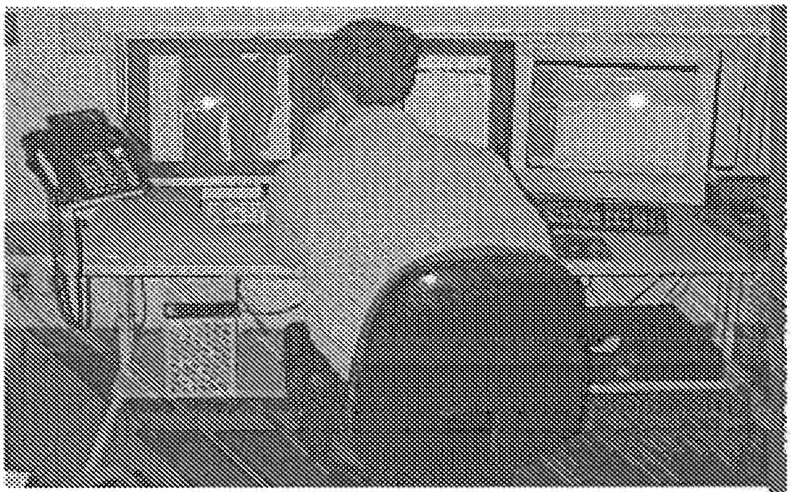
- Scheduled appointments show directly on the modality via DICOM Modality Worklist.
- Image acquisition, header validation and image transfer into the PACS.
- Primary interpretation by the radiologists using various report generation techniques including back-office voice recognition.

Web Distribution

- Web distribution of the report and images electronically to the original referring physician, and to any clinician in the system that has the appropriate access and the need-to-know this diagnostic information. Images from radiology as well as from the cardiac cath lab and other imaging areas will demonstrate the multidisciplinary aspects of the Brigham PACS program and round out the presentation.

BWH Vendors:

- Agfa HealthCare Corporation
- GE Medical Systems
- Siemens Medical Solutions
- eScription
- Others



Tour 3

CHILDREN'S HOSPITAL OF BOSTON

Children's Hospital Boston is a 325-bed comprehensive center for pediatric health care. As the largest pediatric medical center in the United States, Children's offers a complete range of health care services for children from 15 weeks gestation through 21 years of age (and older in special cases). Children's records approximately 18,000 inpatient admissions each year, and our more than 150 outpatient programs and emergency services care for more than 300,000 patients annually. The hospital also performs 150,000 radiological examinations every year.

The tour includes:

Nuclear Medicine Information System

- See a demonstration of Children's Hospital's "homegrown" NMIS. A brief history and technical overview of the system will be provided followed by a demonstration of the various modules within the NMIS.

Radiology Web Strategy

- Review the Children's Hospital internal radiology website that will highlight the following functional areas in the department: Administrative, Billing, Clinical, Engineering/IT Problem Reporting, PACS Project Tracking System, Radiology IT Information, Research, Staff, Teaching Files/Links, and Training & Conferences.

Technical Tour

- See our newly constructed computer room highlighting our uniquely designed PACS architecture. Radiology IT and PACS vendor personnel will be available to answer questions about how and why we decided on this particular design.
- View our state-of-the-art MRI reading room and view a demonstration of our PACS soft copy reading stations and our VR dictation stations. An opportunity to experience 'hands-on' PACS stations will be provided.

Children's Vendors:

- Fujifilm Medical Systems
- EMC Corporation

Tour 4

NEW ENGLAND BAPTIST HOSPITAL

Established in 1893, New England Baptist Hospital is a 140-bed adult medical/surgical hospital, located in the Mission Hill neighborhood of Boston, with specialty services in musculoskeletal care, sports medicine, occupational medicine and cardiology. Since its inception, New England Baptist Hospital has continually taken patient care to new levels and today is recognized for its exceptional blend of caring and commitment.

The New England Baptist Hospital Department of Radiology is a comprehensive diagnostic entity, encompassing general radiology, CT, MRI, ultrasound, mammography, nuclear medicine and PET.

New England Baptist Hospital Radiology Department was the first electronically integrated facility in Boston. NEBH Radiology takes in electronic images from 3 satellite locations and currently is archiving 85,000 exams per year. The reading room consists of 12 diagnostic workstations as well as 3 dedicated for clinical review. Image distribution throughout the campus and clinicians offices is handled via a dedicated web server. At present more than 200 users access patient image data and radiologist interpretation via Web 1000.

The tour includes:

- A hands-on look at workflow in the surgical suites, using 42" plasma screen technology in the OR setting. With emphasis toward joint revision, we will display the newest advances in electronic templating.
- Showcase of workflow in clinic settings in dedicated Hand Surgical and Outpatient area.
- Available images printed on paper media using DICOM print.
- Demonstration of primary interpretation of PACS stations, using PACS/RIS/MIS interface.

NEB Vendors:

- Agfa HealthCare Corporation
- RTAS Systems
- Meditech

Tour 5

MASSACHUSETTS GENERAL HOSPITAL

Founded in 1811, the Massachusetts General Hospital (MGH) is the third oldest general hospital in the United States and the oldest and largest in New England. The 868-bed world-renowned medical center offers sophisticated diagnostic and therapeutic care in virtually every specialty and subspecialty of medicine and surgery. Each year the MGH admits approximately 42,000 inpatients and handles more than 1.2 million visits in its extensive outpatient programs at the main campus and at its four health centers. Its emergency services handle nearly 75,000 visits annually.

The MGH conducts the largest hospital-based research program in the United States, with an annual research budget of more than $300 million. It is the oldest and largest teaching hospital of Harvard Medical School, and nearly all of the hospital's active staff physicians are on the Harvard Medical School faculty.

There are approximately 30 million radiology images stored in the PACS system at MGH making it the largest PACS System in the world. The radiology department performs an average of 1400 exams per day and about 450,000 radiological exams each year. There are 7 Interpretation Locations: Pediatrics, Neuroradiology, Bone, Chest, Gastrogenitory, Emergency Radiology and Vascular. With a staff of over 70 board-certified radiologists, and an exceptionally high volume of studies, the department has gained distinction for its subspecialty expertise in cardiac, emergency, GI/GU, interventional, musculoskeletal, neurology, interventional neurology, pediatric, thoracic, and vascular radiology, as well as breast imaging and nuclear medicine.

The tour includes:

Digital Imaging Department

- See where it all happens: System Monitoring, QA, Troubleshooting.

Emergency Department

- See CR, DR, CT functioning in a Level 1 Trauma Emergency Department.

Orthopaedic Outpatient Department

- See CR, DR functioning in a busy orthopaedic radiology department.

Interpretation Areas

- Observe primary interpretation using PACS system in conjunction with voice recognition.

The Image Service Center

- See the state-of-the-art Image Service Center.

MGH Vendors:

- Agfa HealthCare Corporation
- Amicas, Inc.
- GE Medical Systems
- Hologic, Inc.
- Siemens Medical Solutions

Exhibitors

Companies that have confirmed their participation at SCAR 2003 as of publication time.

ADVANCE Newsmagazines
Agfa HealthCare Corporation
AMICAS, Inc
Aware, Inc.
BarcoView, LLC
BRIT Systems, Inc.
Cambridge Computer Services, Inc.
Canon Medical Systems
CCA (Creative Computer Applications, Inc.)
Cerner Corporation
Codonics, Inc.
Data Distributing, LLC
Data-Ray Corporation
DatCard Systems, Inc.
Decisions in Imaging Economics
DeJarnette Research Systems, Inc.
Diagnostic Imaging
Dictaphone
Eastman Kodak Company
eDictation
Emageon Inc.
eMed Technologies Corporation
eRAD/Image Medical
Fujifilm Medical Systems USA, Inc.
GE Medical Systems
Hologic, Inc.
iDX Systems Corporation
Image Systems Corporation
Images-on-Call
Intelerad Medical Systems
Konica Medical Imaging
McKesson Information Systems
Medical Manager Health Systems
MEEN Imaging Technology News
Merge eFilm
Misys Healthcare Systems
NAI Technology Products
Orex Computed Radiography
peerVue
Philips Medical Systems
Planar Systems, Inc.
PointDx, Inc.
ProVox Technologies
Quest International, Inc.
R2 Technology, Inc.
RADinfosystems
Redrick Technologies, Inc.
Richardson Electronics
RIS Logic, Inc.
Rorke Data, Inc.
ScImage
Siemens Display Technologies
Siemens Medical Solutions, USA, Inc.
SmartPACS
Softmed Systems, Inc.
Sorna Corporation
Source Medical Solutions
Springer-Verlag New York, Inc.
Stentor
StructureRad LLC
Tech Source
Titan Systems Corporation
Tourism Vancouver
U.S. Electronics, Inc./Totoku
U.S. Radiology On-Call
UltraVisual Medical Systems
VIDAR Systems Corporation
Vital Images, Inc.
VitalWorks
Voxar, Inc.
Witt Biomedical
Ximis, Inc.

SCAR Corporate Members

Acuson Corporation
Agfa HealthCare Corporation
Amicas, Inc.
BarcoView, LLC
Canon Medical Systems
CCA (Creative Computer Applications, Inc.)
Cerner Corporation
DatCard Systems, Inc.
DeJarnette Research Systems, Inc.
Eastman Kodak Company
Emageon, Inc.
eMed Technologies Corporation
First Consulting Group
Fujifilm Medical Systems USA, Inc.
GE Medical Systems
Hologic Inc.
IDX Systems Corporation
IMV Medical Information Division, Inc.
Johnson & Baughan, P.A.
Konica Medical Imaging, Inc.
McKesson Information Systems
Medical Technology Services (MTS)
Merge eFilm
Misys Healthcare Systems
Packeteer
Philips Medical Systems, NA
R2 Technology, Inc.
SG&A Consulting, Inc.
Siemens Medical Solutions, USA, Inc.
SmartPACS
Stentor
Tech Source, Inc.
Toshiba America Medical Systems
UltraVisual Medical Systems
Vidar Systems Corporation

Corporate Sponsors

SCAR wishes to thank the following corporations for their generous support of the SCAR 2003 annual meeting.

Platinum Level Sponsorship

Siemens Medical Solutions, USA, Inc.

GE Medical Systems

Gold Level Sponsorship

Eastman Kodak Company

Silver Level Sponsorship

Emageon, Inc.

Fujifilm Medical Systems USA, Inc.

IDX Systems Corporation

Visit the SCAR 2003 Exhibits

Two full days of exhibits held in Hall A at the Hynes Convention Center feature industry leaders demonstrating the latest products and services in medical imaging, informatics, and information technology. More PACS vendors assemble at SCAR than any other conference except the RSNA.

Internet terminals and wireless kiosks will be located in the exhibit hall. Lunch and breaks will also be served in the exhibit hall on Sunday and Monday.

Exhibit Hall Hours

Saturday	Opening Reception in Exhibit Hall	5:00 pm to 7:00 pm
Sunday	Exhibit Hall Open	9:30 am to 5:00 pm
Monday	Exhibit Hall Open	9:30 am to 5:00 pm

GE Medical Systems Hosts the Opening Reception in the Exhibit Hall

GE Medical Systems is the official sponsor of the Opening Reception scheduled for Saturday, June 7 from 5:00 pm to 7:00 pm in Exhibit Hall A of the Hynes Convention Center, the site for SCAR 2003 Technical exhibits. 2000+ attendees will enjoy cocktails and hors d'oeuvres and gather to network in the sold out exhibit hall. A not-to-be-missed event!

GE Medical Systems
Information Technologies

Siemens Hosts the SCAR 2003 Top of the Hub — Prudential Tower Reception

Siemens is the generous sponsor of this year's Welcome Reception for all meeting attendees scheduled for Sunday, June 8. The reception will be held from 6:00 pm to 8:00 pm at the top of the Prudential Tower adjacent to the Hynes Convention Center and Sheraton Boston Hotel in the Top of the Hub restaurant. Soaring fifty-two stories above Boston, you can't help but be inspired by the finest of sunsets and the breathtaking views of the entire city.

A highlight of the reception is the presentation of poster awards and cash prizes and a special welcome to new individual, institutional, and corporate members of SCAR.

SIEMENS

Registration Information

Registration Fees

Payments must be made in U.S. dollars by personal check, travelers check, VISA, MasterCard, AMEX or Discover.

Payment in full is required to process your registration.

Cancellation/Refund Policy

All cancellations and requests for refunds must be in writing and received no later than May 16, 2003. Refunds are subject to an $80 administrative fee. No refunds will be issued after May 16, 2003.

Registration Categories

SCAR Member; Non-Member; New SCAR Member: Includes badge, final program, Conference Proceedings, SCAR U syllabus, all sessions, entrance into the exhibit hall. Non-members may elect to add their first year of membership for a discounted fee by registering in the New SCAR Member category.

Daily: Includes badge, final program, Conference Proceedings, SCAR U syllabus, all sessions, entrance into the exhibit hall. SCAR membership may be purchased separately.

Residents/Medical Students Only: Same materials as Non-member. Documentation of student status is required. SCAR membership may be purchased separately for $100.

Spouse/Companion: Registration is complimentary, if the individual is not a member, potential member, or speaker. The Conference Proceedings and SCAR U syllabus are not included.

PACS Administration Course and SCAR affiliated User Group Meetings have additional registration fees.

Official Attire at SCAR 2003 —
Business Casual

SCAR
2003

Registration Form

SCAR 2003 Annual Meeting • June 7–10, 2003 • Boston, Massachusetts
Pre-registration Deadline: May 30, 2003

Occupation
(please select ONE category–best match)

□ Physician
□ Healthcare Administrator (includes CIOs, CEOs, CFOs)
□ Computer Scientist
□ Engineer
□ Health Information Technology Professional
□ Scientist/Researcher
□ Medical Physicist
□ PACS Administrator
□ Technologist
□ Vendor
□ Consultant
□ Other ....

Primary Occupational Setting
(please select ONE category–best match)

□ University Hospital
□ Government or VA Hospital
□ Community Hospital
□ Private Practice (office, clinic or imaging center)
□ Corporate
□ Government (non-hospital)
□ Resident/Medical Student
□ Other ....

Medical Specialty

□ Radiology
□ Cardiology
□ Nuclear Medicine
□ Information Systems
□ Other ....

Meeting Groups
(please check ALL that apply)

□ IRISS Member
□ APUG Member
□ SCAR 2003 Scientific Presenter
□ SCAR 2003 SCAR U Faculty
□ SCAR 2003 Invited Speaker (Opening, Closing, Lunch, and Special Sessions)

How did you learn of the SCAR 2003 meeting?

□ Colleagues
□ Direct Mail
□ SCAR News
□ Journal of Digital Imaging
□ SCAR Website
□ Internet Link (please specify)
....
□ Diagnostic Imaging
□ Other Publication (please specify)
....

Do you plan to attend the SCAR Reception at the "Top of the Hub," Prudential Building on Sunday, June 8th?

□ Yes □ No

Spouse/Companion will attend:

□ Yes □ No

....
FIRST NAME MIDDLE NAME LAST NAME

....
DEGREE TITLE

....
DEPARTMENT

....
ORGANIZATION/INSTITUTION

....
MAILING ADDRESS SUITE/APT

....
CITY STATE ZIP

....
COUNTRY (IF NOT USA)

....
PHONE NUMBER FAX NUMBER

....
E-MAIL ADDRESS

□ My Spouse/Companion will attend (name for badge) ....

Request for CME Credit

Please check the type of credit you wish to receive:
□ Physicians – Category I Credits for CME are offered to physicians.
□ Technologists – Category A Credits for CME offered to radiologic technologists.
□ Physicists – MEPS credits for CME are offered to medical physicists.

Cancellation/Refund Policy

All cancellations and requests for refunds must be in writing and received no later than May 16, 2003. Refunds are subject to an $80 administrative fee. No refunds will be issued after May 16, 2003. Please return both sides of registration form.

Payment:

□ Check enclosed in U.S. Dollars to: SCAR 2003
□ Credit Card: □ VISA □ MasterCard □ AMEX □ Discover

....
CREDIT CARD NUMBER CREDIT CARD EXPIRATION DATE (MM/YY)

....
ZIP CODE OF CREDIT CARD BILLING ADDRESS - U.S. ONLY AUTHORIZING SIGNATURE

Registration Fees	Early Bird Until 5/2/03	After 5/2/03 and On Site	AMOUNT
SCAR Annual Meeting			
SCAR Member Rate	$445	$495	....
Non-Member Rate	$545	$595	....
New SCAR Applicant* Rate	$570	$620	....
Resident/Medical Student Rate	$300	$350	....
Daily (per day—check day(s) below)			
Saturday, June 7	$200	$250	....
Sunday, June 8	$200	$250	....
Monday, June 9	$200	$250	....
Tuesday, June 10	$200	$250	....
SCAR Pre-Meeting Course, Friday, June 6			
PACS Administration 1-Day Course	$100	$100	....
SCAR Individual Membership Renewal (12 months)			
□ Domestic (USA)	$125	$125	....
□ International (including Canada)	$200	$200	....
□ Medical Student/Resident (USA only)	$100	$100	....
□ Emeritus	$100	$100	....

*Meeting Registration and 1st year SCAR Membership (see inside back cover for SCAR Membership Benefits)

TOTAL AMOUNT ENCLOSED $ ....

One registration per form; copy as necessary.

continues on back

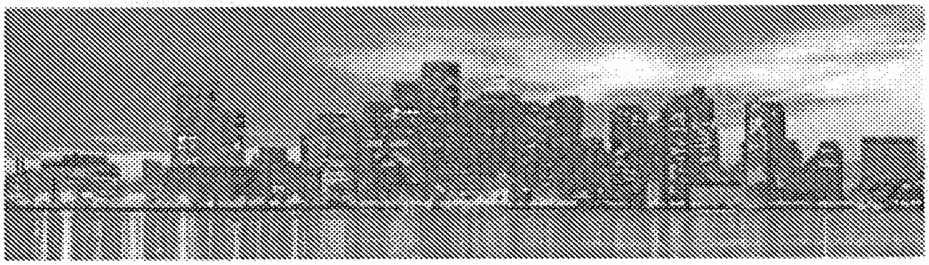
Hospital Tour Registration

NAME ....

The following tours require advance registration. There is no additional fee, but space is limited. You may pre-register for two tours. See page 20 for description of tours at Beth Israel Deaconess Medical Center (BID), Brigham and Women's Hospital (BWH), Children's Hospital of Boston (CHB), Massachusetts General Hospital (MGH), and New England Baptist Hospital (NEB).

	TOUR 1 Beth Israel Deaconess	TOUR 2 Brigham and Women's	TOUR 3 Children's of Boston	TOUR 4 New England Baptist
Saturday, June 7				
1:15 PM	Tour 1A – BID	Tour 2A – BWH	Tour 3A – CHB	Tour 4A – NEB
3:15 PM	Tour 1B – BID	Tour 2B – BWH	Tour 3B – CHB	Tour 4B – NEB
Sunday, June 8				
1:15 PM	Tour 1C – BID	Tour 2C – BWH	Tour 3C – CHB	Tour 4C – NEB
3:15 PM	Tour 1D – BID	Tour 2D – BWH	Tour 3D – CHB	Tour 4D – NEB

	TOUR 5 Massachusetts General
Monday, June 9	
1:15 PM	Tour 5E – MGH
3:15 PM	Tour 5F – MGH



How many hospital tours would you like to attend?

Please Circle: 1 2 None

Select your Hospital Tour by number and letter code*

(Example: "1A" for Beth Israel Deaconess at 1:15 pm on Saturday, June 7)

1st Choice ....

2nd Choice ....

3rd Choice ....

4th Choice ....

5th Choice ....

* Your tour times will be on your registration confirmation. Tour tickets and instructions will be in your registration packet.

Americans With Disabilities Act

Individuals needing auxiliary aids or services as identified in the Americans with Disabilities Act, please call the Society for Computer Applications in Radiology at (703) 757-0054.

Three Easy Ways to Register

Internet: www.scarnet.org (Credit Card Only)

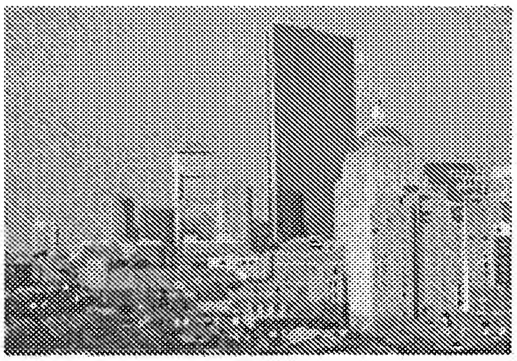
Fax: 703-757-0454 (Credit Card Only)

Mail: SCAR 2003 Meeting Registration
10105 Cottesmore Court
Great Falls, VA 22066-3540

Allow up to 3 weeks for receipt of your registration confirmation letter.

Keep a copy of this form for your records.
Please return both sides of registration form.

Hotel and Travel Information



SCAR is very excited to hold their 2003 Annual Meeting in downtown Boston, Massachusetts. This location enables everyone to learn from the "Boston experience," with special sessions taught by radiology informatics faculty of Boston medical schools and tours of electronic imaging activities at leading Boston healthcare facilities.

For more information on Boston, visit the Greater Boston Convention and Visitors Bureau at www.bostonusa.com

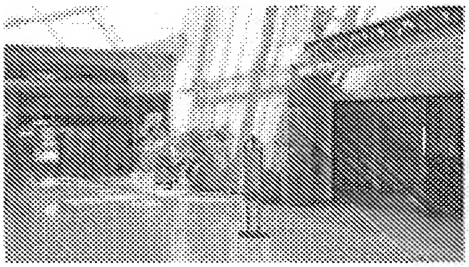
HOUSING DEADLINE: APRIL 28, 2003

SHERATON BOSTON HOTEL

Group rates are available at the Society for Computer Applications in Radiology headquarter hotel – the Sheraton Boston Hotel. The hotel is connected to the Hynes Convention Center. All scientific and educational sessions will be held in the Sheraton Boston Hotel. Technical exhibits will be located in the Hynes Convention Center.

The 1,215-room Sheraton Boston, New England's largest hotel, has recently completed a full-scale renovation project, positioning it as one of the region's premiere business and convention venues. Nestled in charming and historic Back Bay, the Sheraton Boston Hotel is 4 miles from Boston's Logan International Airport. The Sheraton is close to the Financial District and businesses in Copley Square and Downtown Crossing, and one block from famed Newbury Street, the scenic Charles River and many favorite shops, restaurants and museums.

The hotel is connected via an indoor walkway to the Hynes Convention Center and to two hundred shops at the Prudential Center and Copley Place Mall. Saks Fifth Avenue, Ann Taylor, Gucci, Neiman Marcus, and Williams-Sonoma are just some of the fine retail establishments within this expansive complex.



ROOM RESERVATIONS

Room reservations can be made at the Sheraton Boston by calling **800-325-3535** or by faxing the attached housing form no later than Monday, April 28, 2003. After this date, reservations will be accepted on a space available basis at the SCAR meeting rate. Reservations should be made **directly with the hotel, not with SCAR.**

Some important information when booking your reservation:

- Please make reservations early. Cut off date for room reservations is Monday, April 28.
- Be sure to tell the Sheraton you are with SCAR or Society for Computer Applications in Radiology to receive the discounted room rate. Discount rate also applies for attendees participating in the IRISS, Fuji, and APUG user group meetings, and the PACS administration course.
- Ask the Sheraton to send you a written confirmation.
- The hotel will refund deposit if cancellation of reservation is received 72 hours prior to arrival date.

AIRLINE RESERVATIONS

Discounted fares are available through United Airlines and American Airlines. For United, call 800-521-4041 and refer to Meeting ID Code 511RP. For American, call 800-433-1790 and refer to Starfile #3863AO.

AmericanAirlines

UNITED AIRLINES

GROUND TRANSPORTATION

The Sheraton Boston Hotel is 4 miles from Logan International Airport. Back Bay Coach is available from Logan Airport to the Sheraton Boston at a cost of $9.00 one way. Once you have claimed your luggage, call 888-222-5229 for pickup arrangements. Shuttle departs outside the baggage claim area approximately every 20 minutes between 7:00 am and 7:00 pm. Taxi service is also available at an approximate cost of $30.00 one way for up to four people. Other transportation options include subway and commuter rail (MBTA), and Amtrak. See the SCAR Website for driving directions and maps.

PARKING

Valet overnight parking at the hotel is approximately $33.00 per day. There are additional parking garages nearby where you can self-park. Prudential Garage is $32.00, and for the Pilgrim Parking Garage, the daily rate is $24.00.

CAR RENTAL

SCAR has arranged discounted rates for car rentals through Hertz. For reservations, call 800-654-2240 and refer to CV #02010008. Or you may contact your travel agent. Attendees may also place their reservations online at www.hertz.com.

Hertz

Hotel Reservation Form

SCAR 2003 Annual Meeting • June 7–10, 2003 • Boston, Massachusetts

SCAR
2003

SHERATON BOSTON HOTEL
Attn: Reservations Department
39 Dalton Street • Boston, MA 02199
800-325-3535 • FAX: 617-236-6095

RETURN BY APRIL 28 to the Sheraton Boston Hotel

PLEASE PRINT OR TYPE

ACCOMMODATION REQUESTS:
SCAR Room Rates Guaranteed June 4th to June 11th.

Arrival Date .... **Time** .... **CHECK-IN TIME IS AFTER 3:00 pm**

Departure Date .... **Time** .... **CHECK-OUT TIME IS BEFORE 12:00 Noon**

□ Single Room/One Bed ....$239.00 per night
□ Double Room/Two Beds .....$239.00 per night
□ Club Floor/One Bed.......$279.00 per night
□ Club Floor/Two Beds ......$279.00 per night

□ Request Smoking □ Request Non Smoking □ Special Needs/Requests: ....

Please reserve accommodations for:

....
NAME

....
INSTITUTION / COMPANY

....
ADDRESS

....
CITY STATE ZIP CODE

....
COUNTRY EMAIL

....
HOME PHONE WORK PHONE
(If outside USA please also include country and city codes.)

....
NAME OF ACCOMPANYING ADULT

Method of Payment:

Advance deposit is required to confirm room reservations. All rates are in USD and subject to a 12.45% room tax.

□ Check enclosed
□ Credit Card
□ American Express □ MasterCard □ Visa □ Discover □ Diners

....
CREDIT CARD NUMBER EXPIRATION DATE (MM/YY)

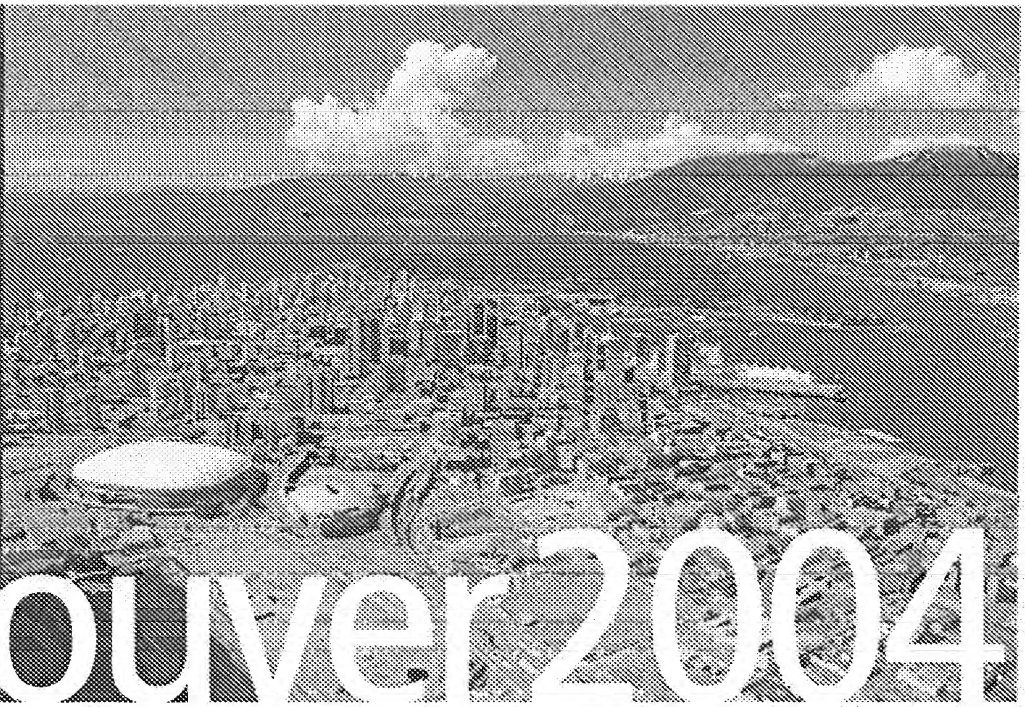
A one night's deposit or credit card number is required in order to guarantee a room reservation. A deposit will be refunded if cancellation of reservation is received 72 hours prior to arrival date. To assure your accommodation, your reservation must be received prior to Monday, April 28, 2003 and before the SCAR reservation block is filled. After this time, reservations will be accepted on a space available basis at the SCAR meeting rate.

Signature: .... Date: ....

Fax or mail to the above address. Keep a copy for your records!

Looking Ahead to SCAR 2004

May 20–23, 2004
Vancouver Convention and Exhibition Centre
Vancouver, British Columbia • Canada



Vancouver 2004

SCAR Membership Information

Special SCAR 2003 Membership Offer

Enjoy the SCAR 2003 conference now and benefit from SCAR membership all year long. SCAR provides an open environment for imaging information professionals to access expert and cutting edge resources in a collegial and practical atmosphere.

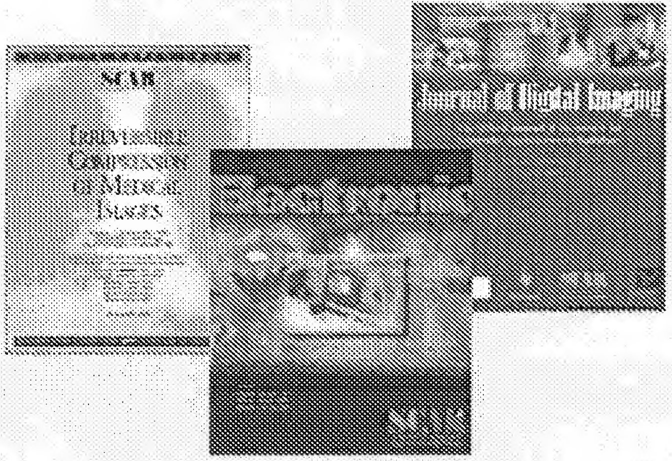
SCAR members are a diverse group of physicians, physicists, technologists, radiology administrators, PACS administrators, CIOs, CFOs, CEOs, IT professionals, engineers, and computer scientists practicing in the academic, government, and private practice sectors. Each constituency contributes its own unique viewpoints, needs, and expertise, creating a stimulating environment for original research, collaboration, and education.

SCAR membership benefits include:

- Subscription to the peer-reviewed *Journal of Digital Imaging*, published quarterly. Now available online to SCAR members.
- The quarterly society update, *SCAR News*.
- Unlimited access to the SCAR email expert hotline, which enables you to ask questions of SCAR's cadre of experts.
- Reduced registration fees for SCAR conferences.
- SCAR U Online member discount.
- Member discounts on SCAR Publications (including the SCAR U Primer Series).
- Opportunities to network with colleagues—the best way to exchange new ideas and concepts.

To begin your membership, just check the NEW SCAR Member box on the registration form or join online by completing an application on the SCAR Website.

SCAR Members—be sure to return your renewal notices to receive the discounted member rate on your SCAR 2003 registration.



SCAR 2003

Society for Computer Applications in Radiology
10105 Cottesmore Court
Great Falls, VA 22066

ANNUAL MEETING 2003

The 20th Symposium for Computer Applications in Radiology

SCAR 2003

JUNE 7–10, 2003 • BOSTON, MASSACHUSETTS

Information Explosion: Embracing Our Future

SCAR 2003 is the leading educational forum with an emphasis on diagnostic image and information interpretation and management in the 21st century

Register early and online at www.scarnet.org

Computer-Assisted Diagnosis: Breast and Thoracic Imaging

Heber MacMahon M.D.
Department of Radiology
The University of Chicago

Dr. MacMahon has received research funding and stock options from R2 Technology, Inc. and research support from: Mitsubishi Space Co., General Electric Medical Systems, Eastman Kodak Co., and Fuji Medical Systems.

macm@midway.uchicago.edu

CAD for Chest Radiographs

Enhancement

- Energy Subtraction
- Temporal Subtraction
- Tomosynthesis

Detection

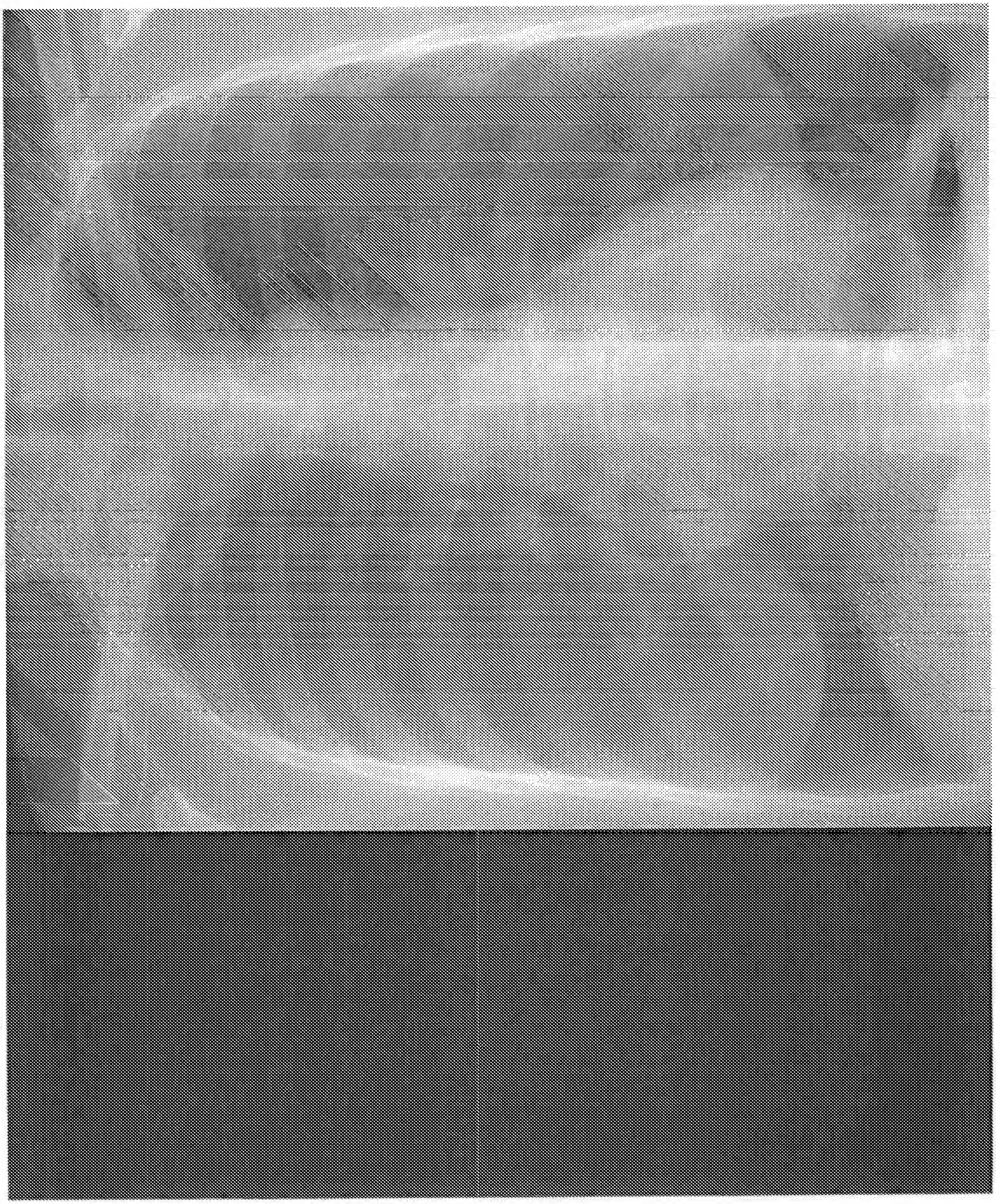
- Nodules
- Interstitial Dis
- Cardiomegaly
- Pneumothorax

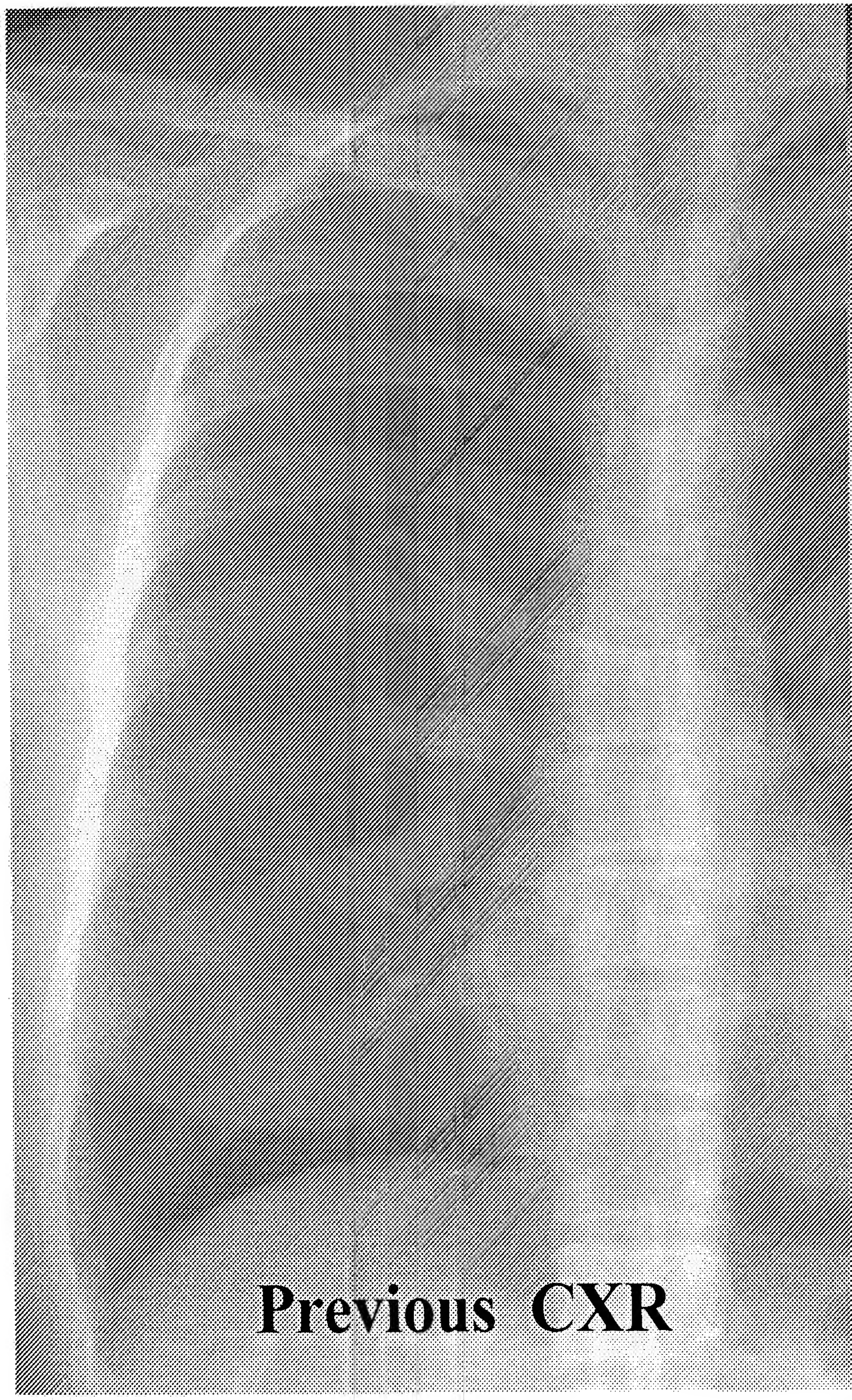
Quantitation

- Interstitial Dis
  - severity
  - nodularity
- Cardiomegaly
  - CT ratio

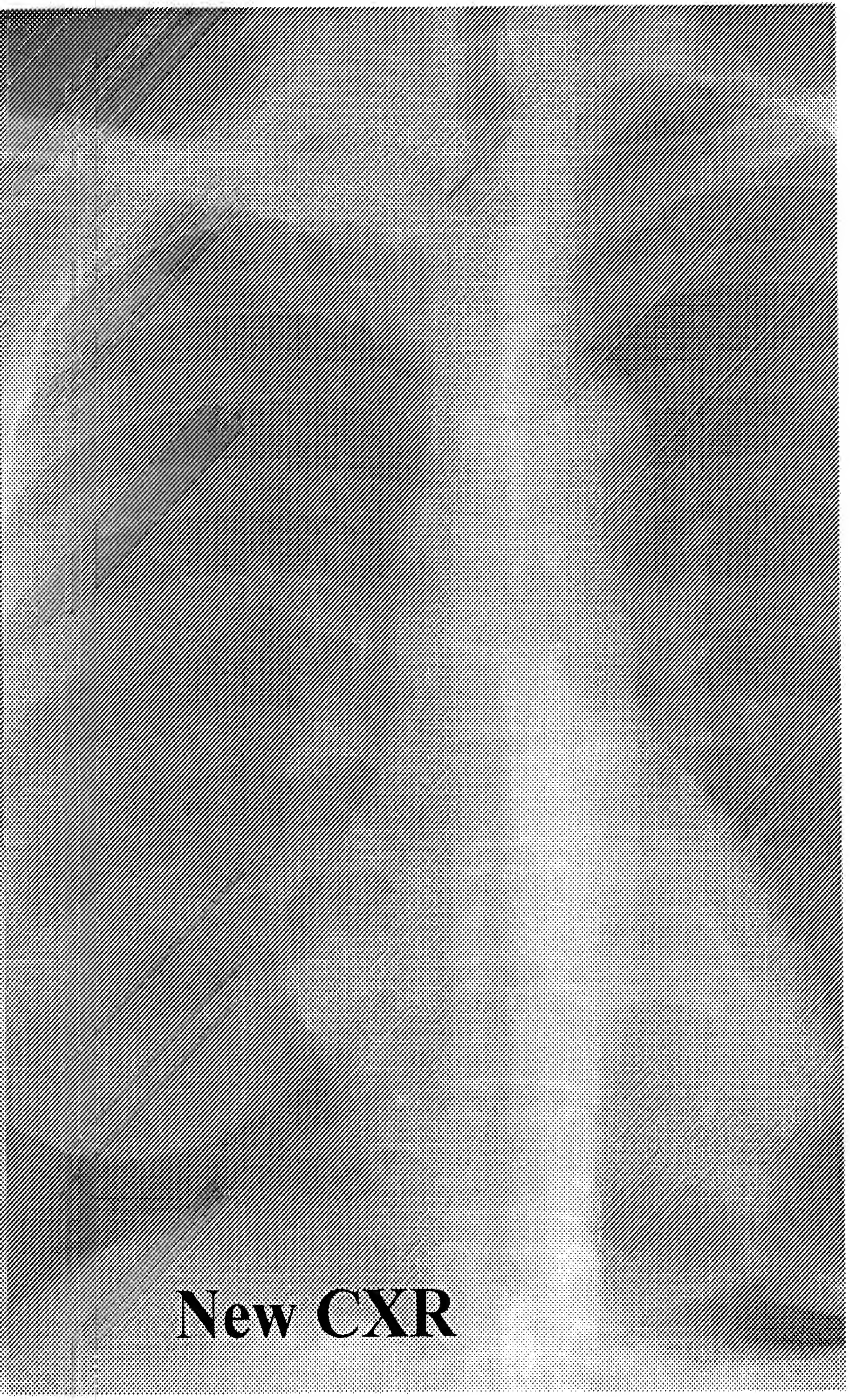
Diagnosis

- ANN for Interstitial Dis
- ANN for Nodules



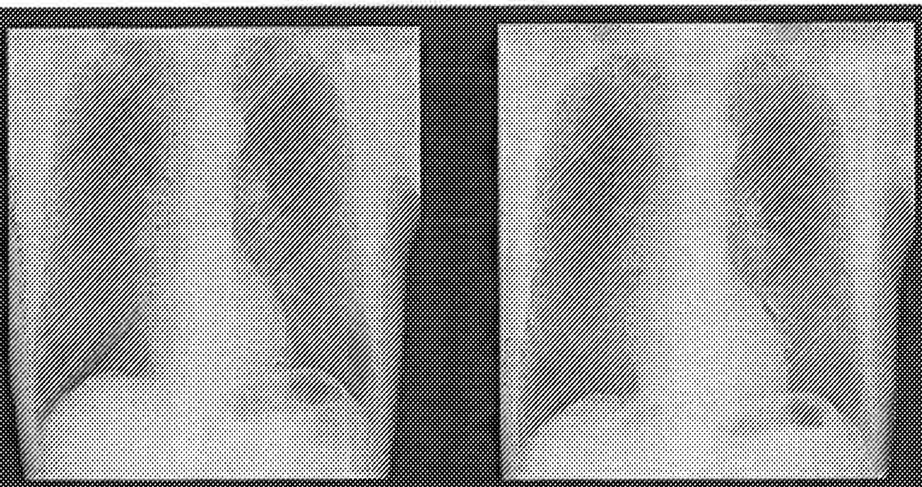


Previous CXR

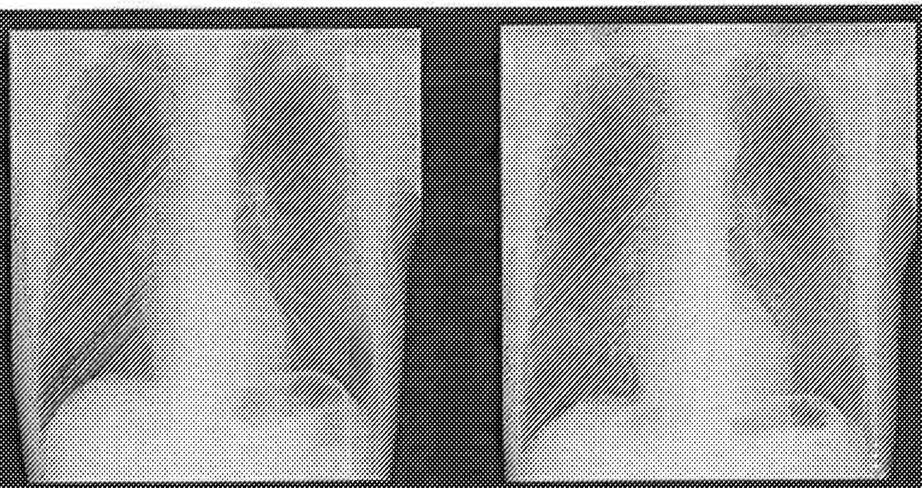


New CXR

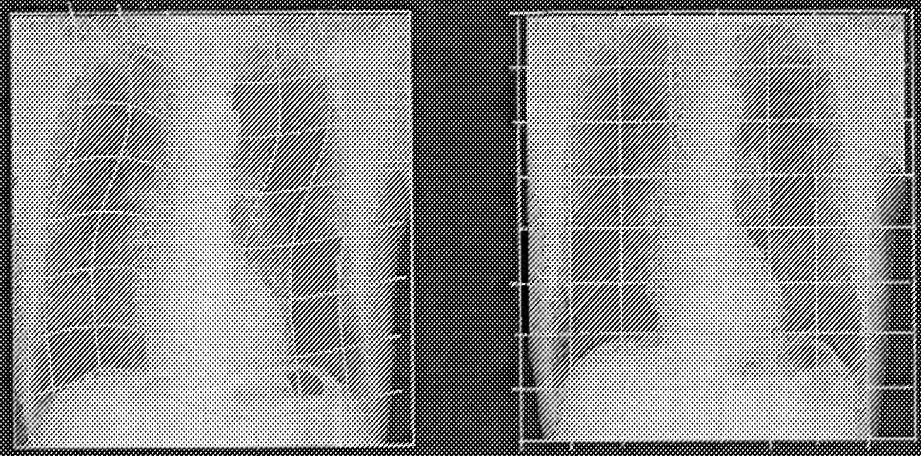
Previous and
Current CXRs

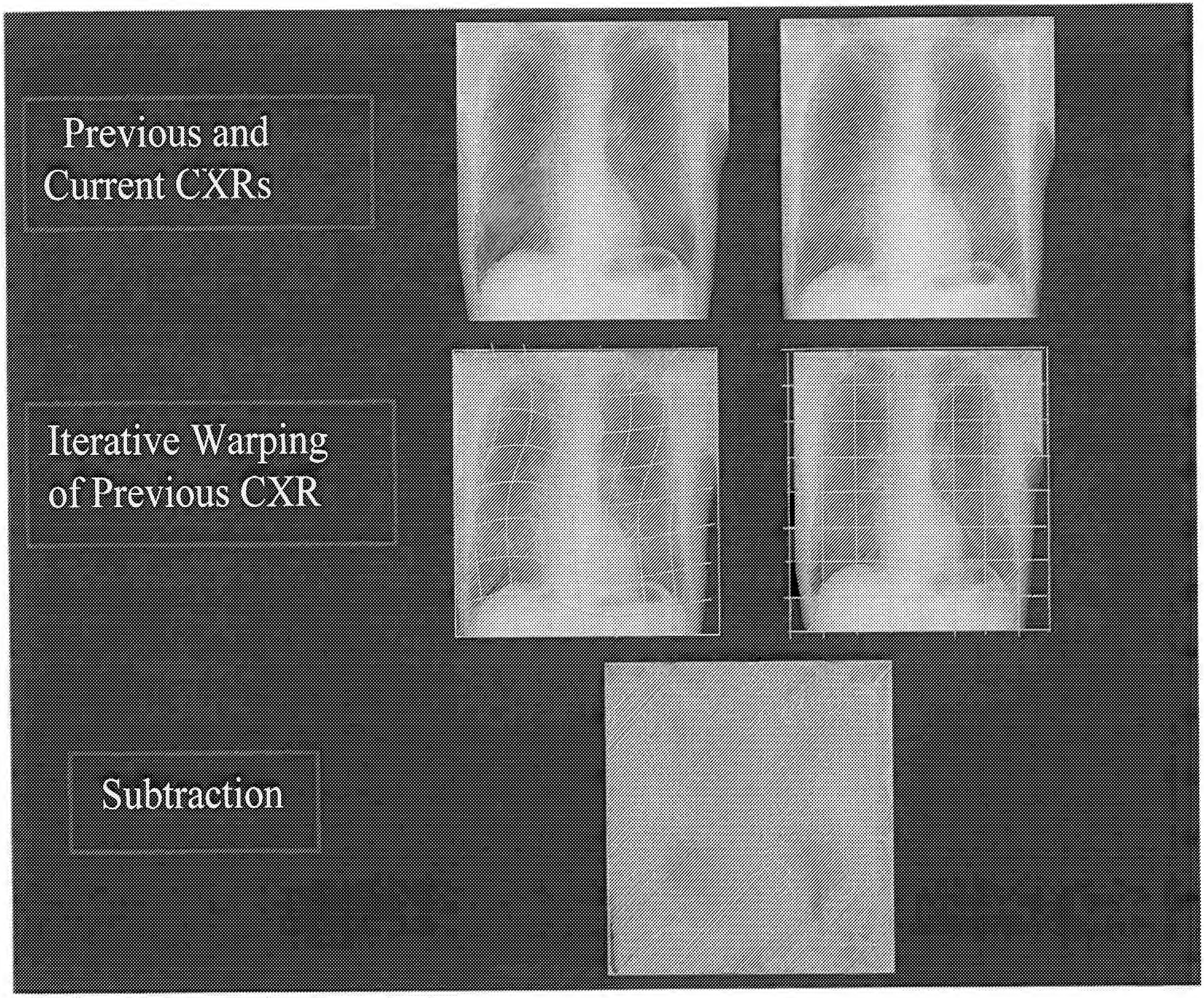


Previous and
Current CXRs



Iterative Warping
of Previous CXR

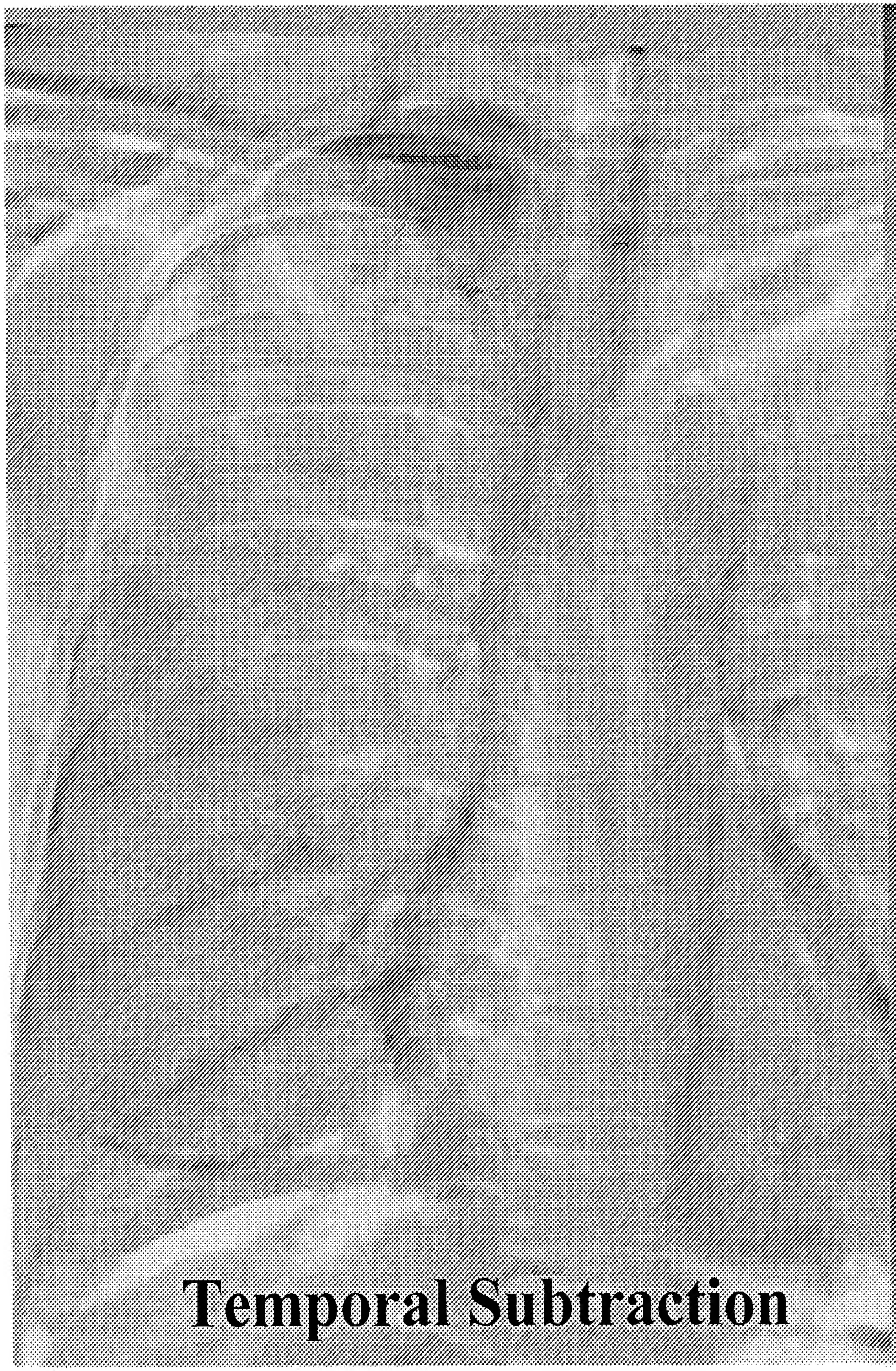




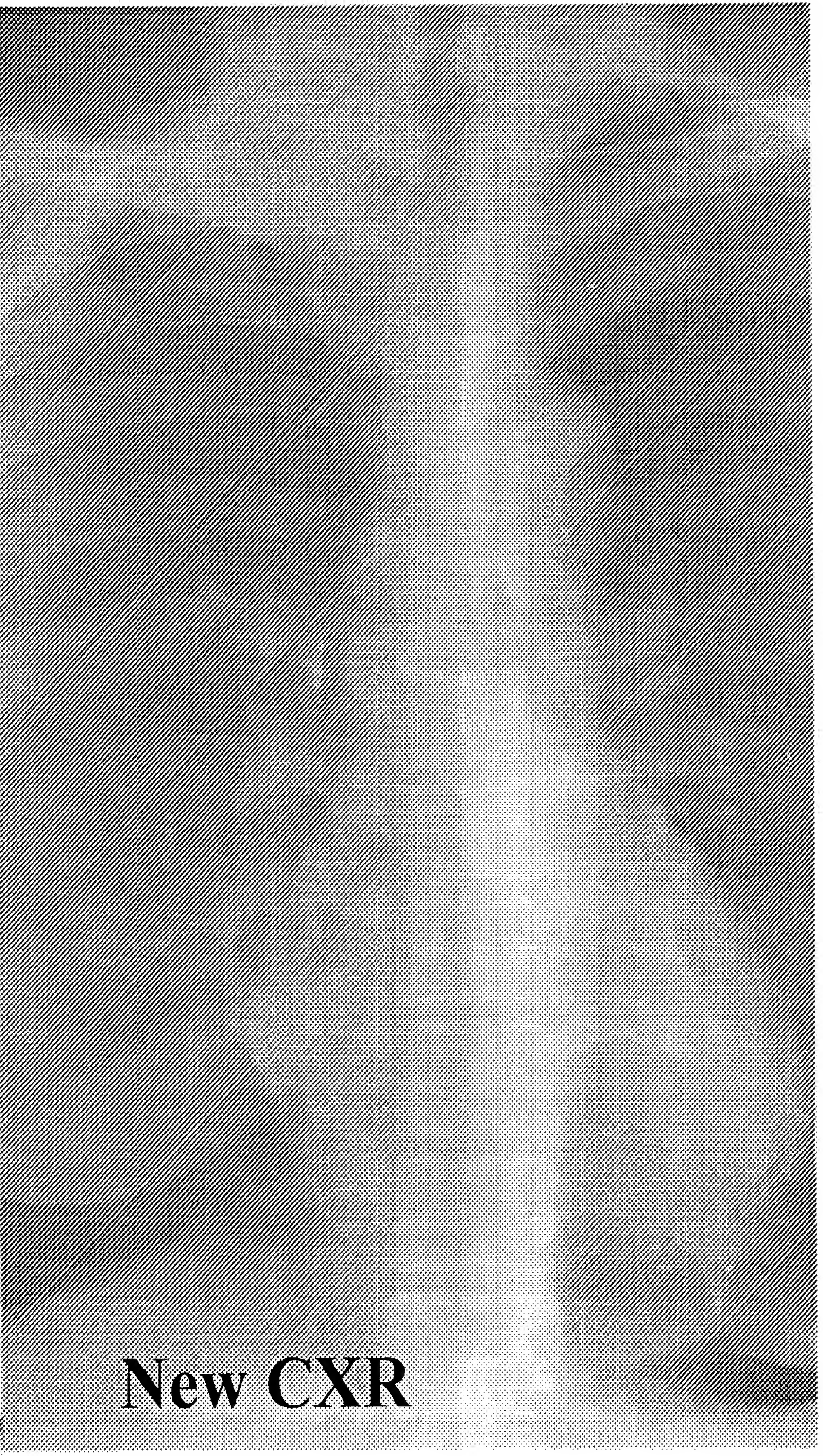
Previous and
Current CXRs

Iterative Warping
of Previous CXR

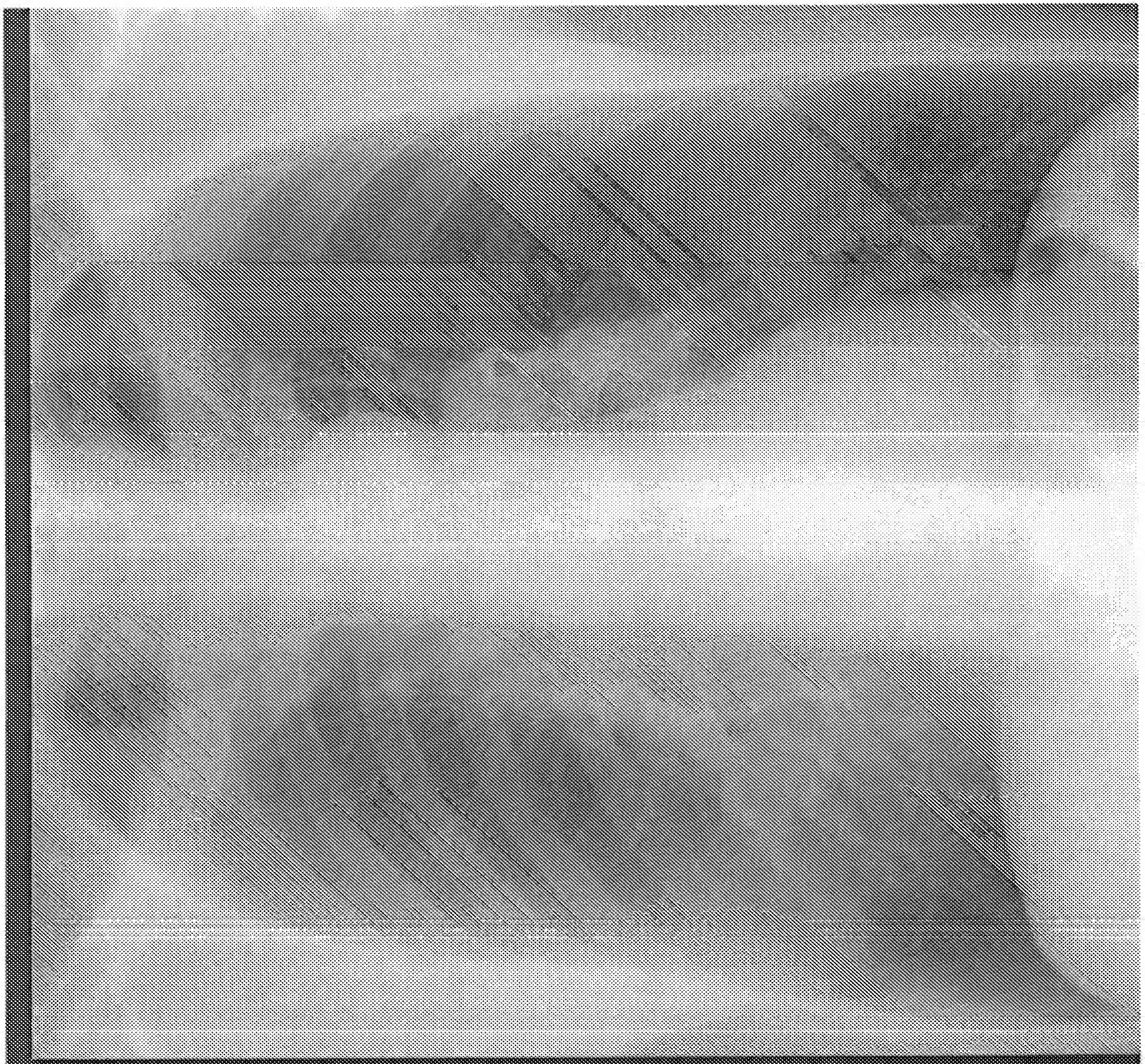
Subtraction

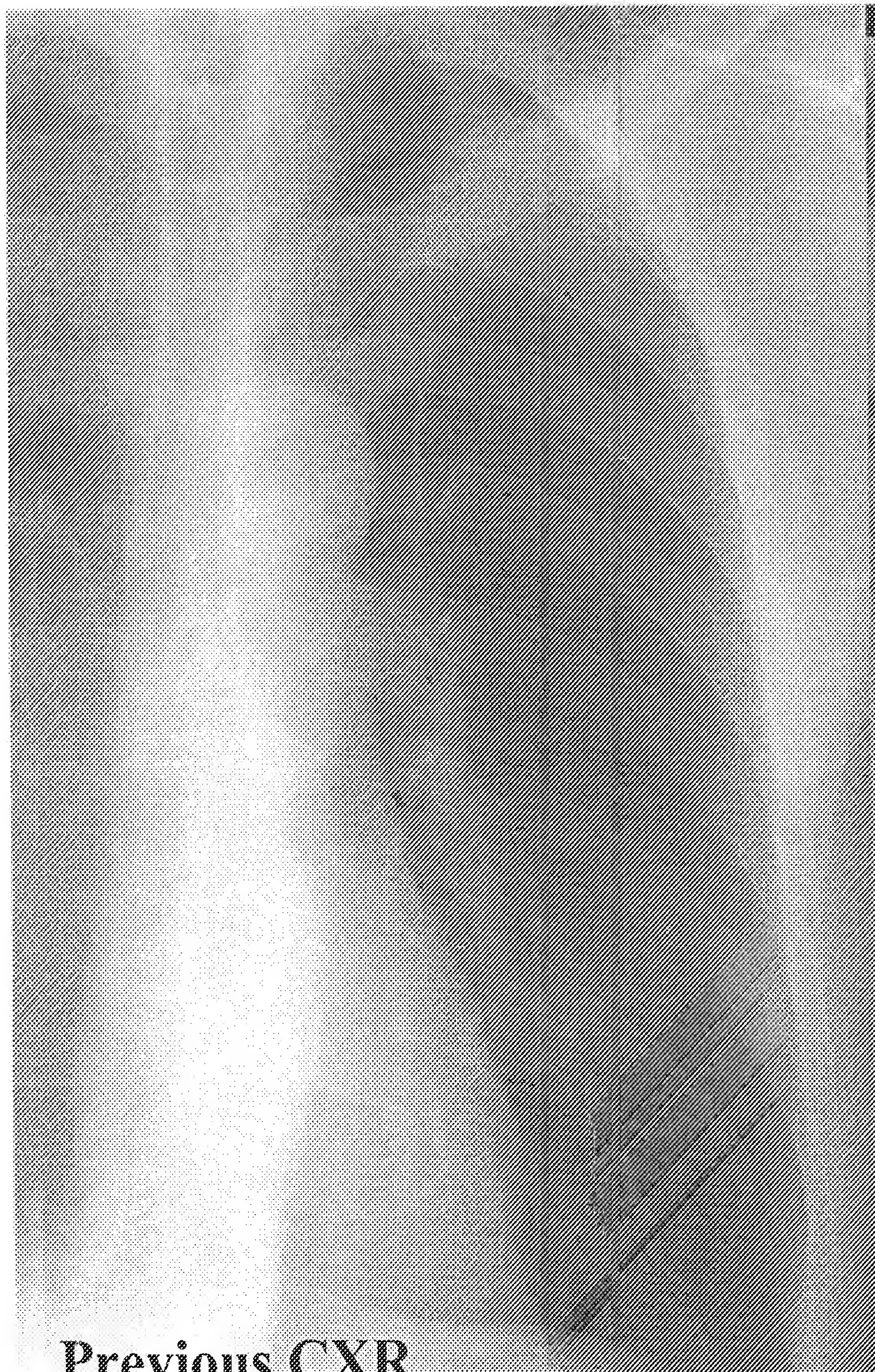


Temporal Subtraction

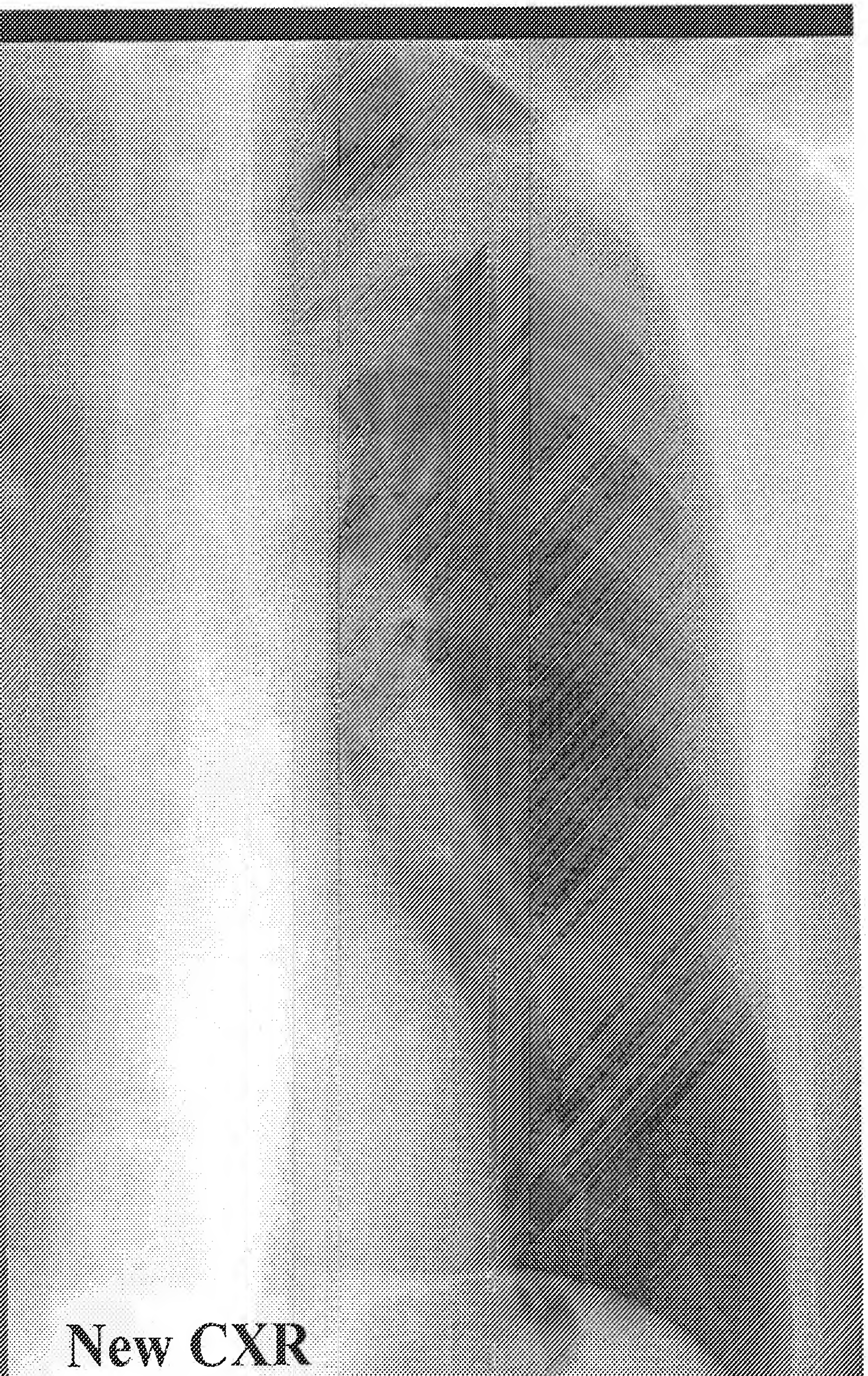


New CXR

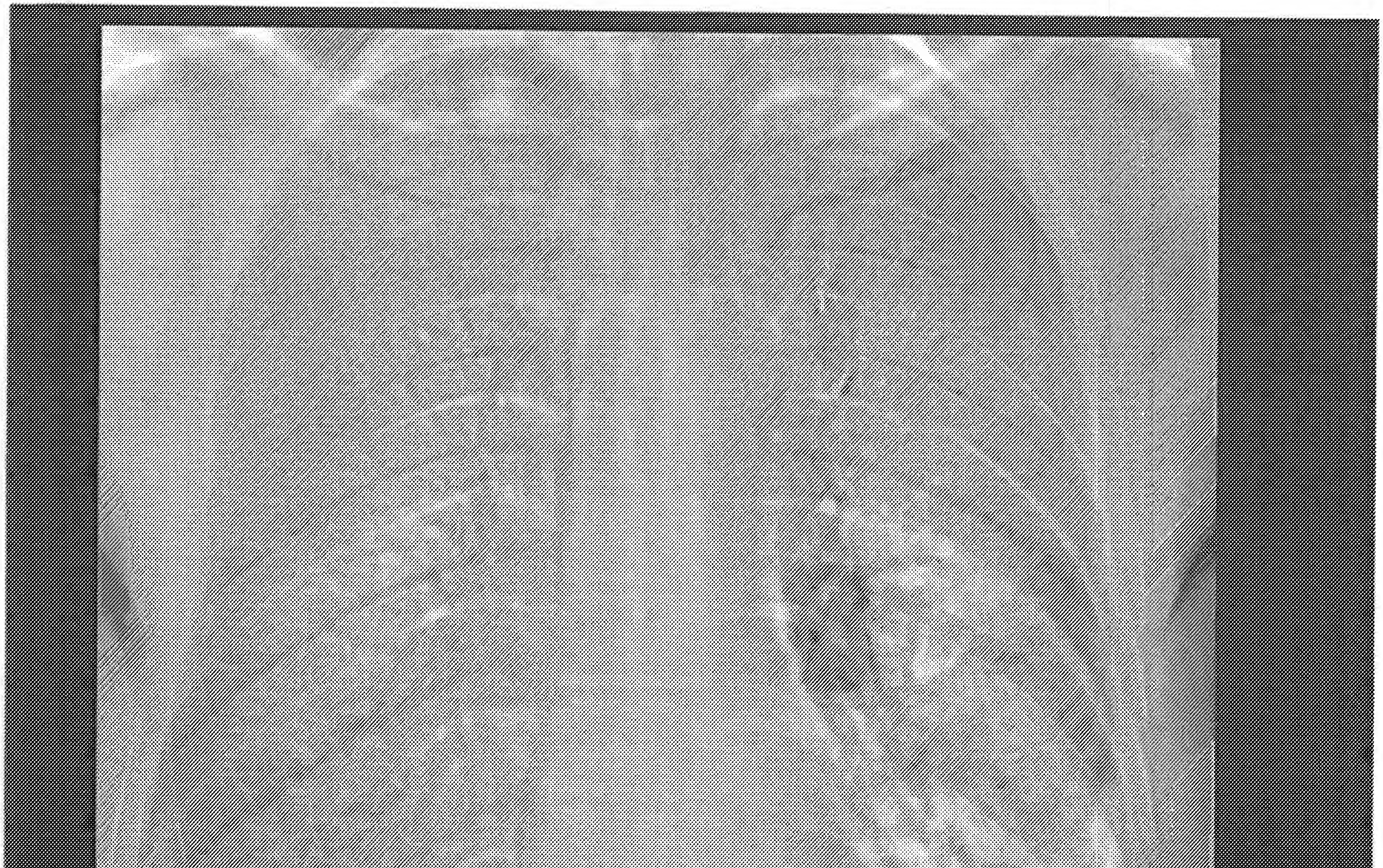




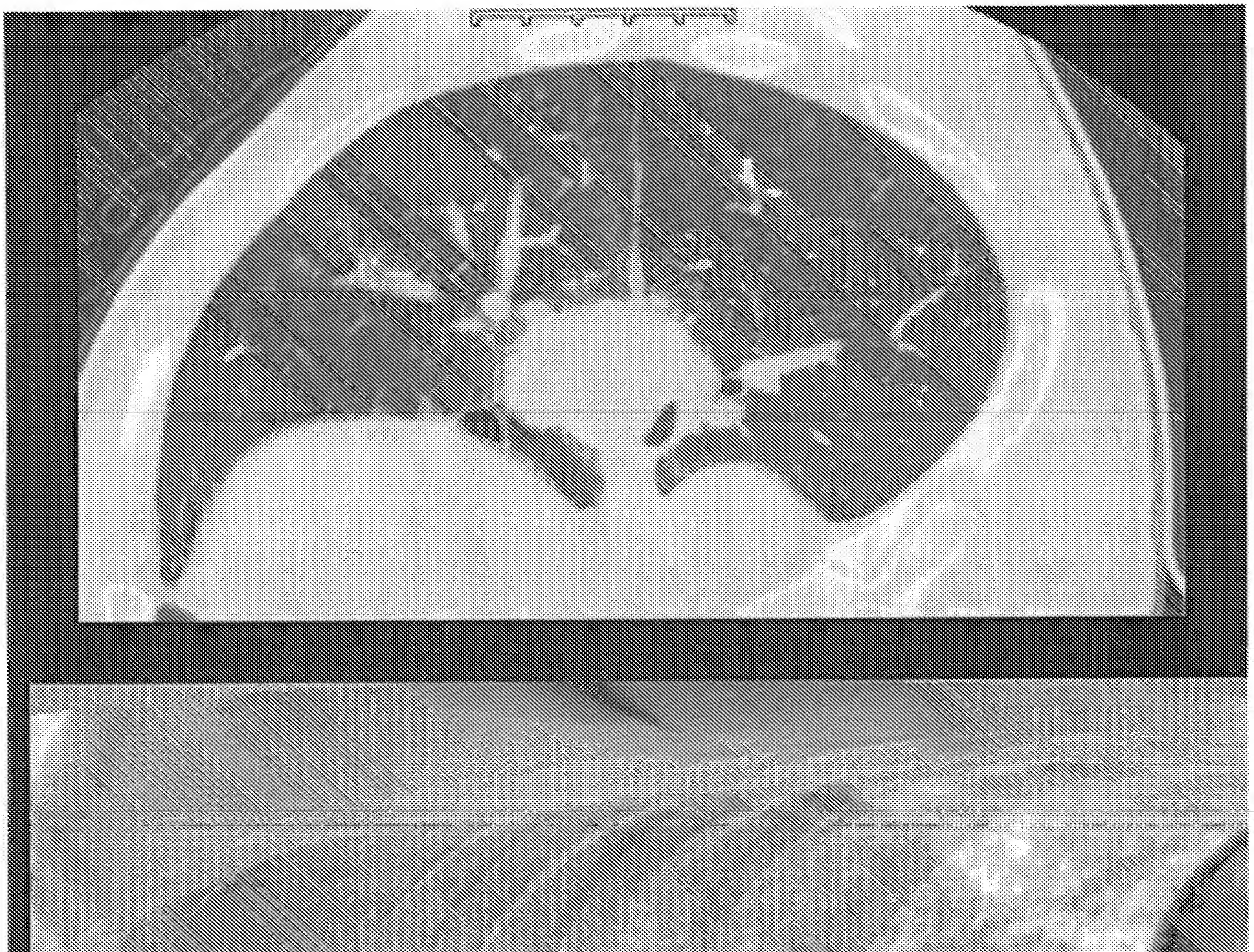
Previous CXR



New CXR



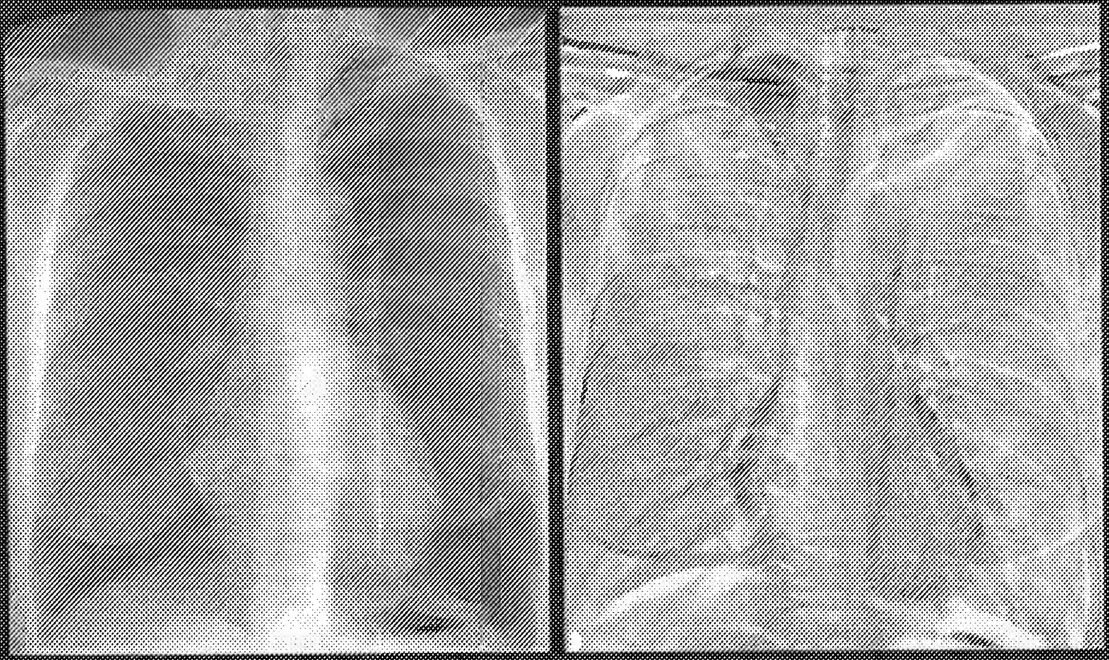
Temporal Subtraction



Temporal Subtraction: Clinical Performance

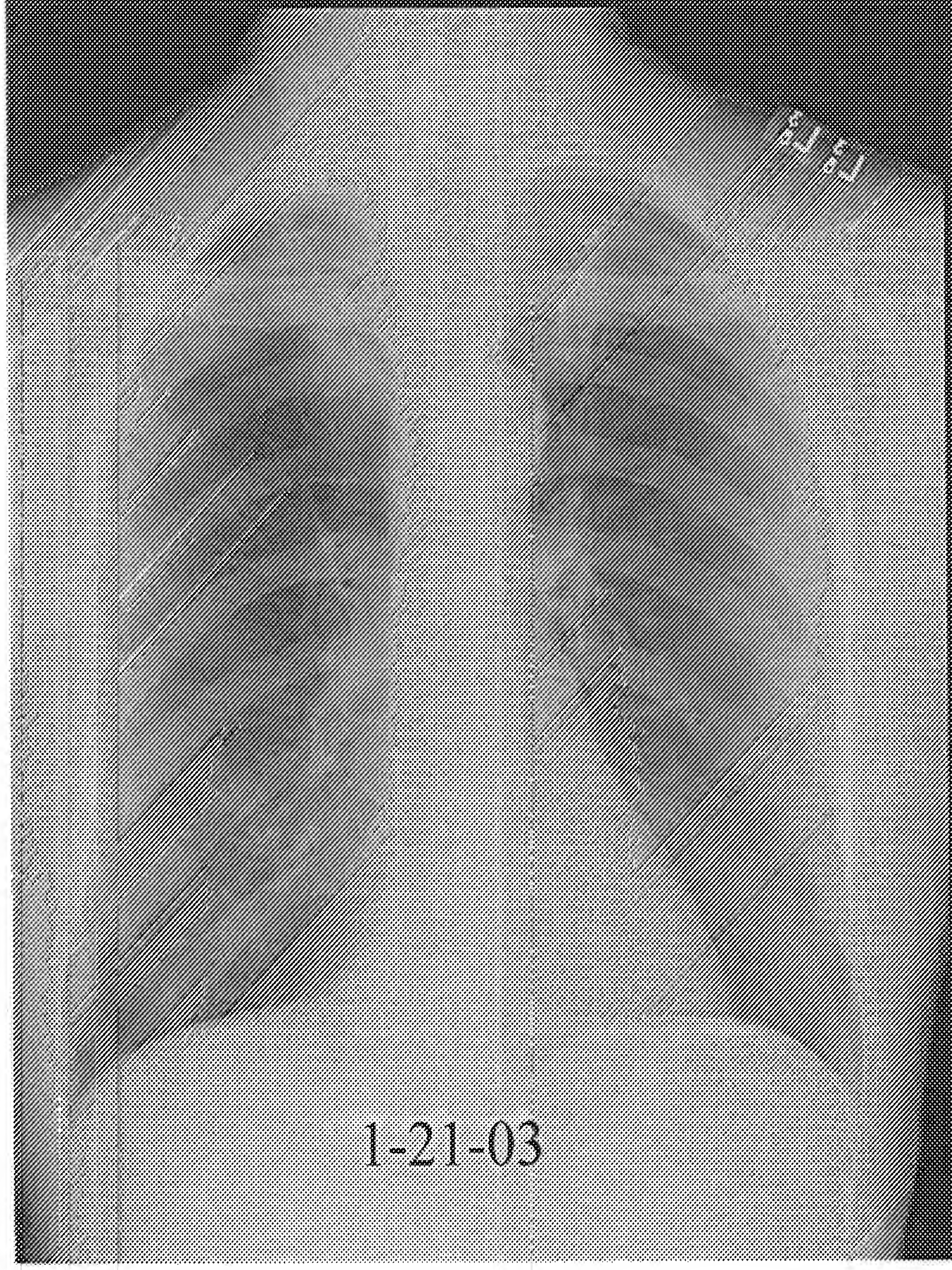
Improved detection:

- low opacity lesions
- obscured nodules
- multifocal disease
- perfusion changes



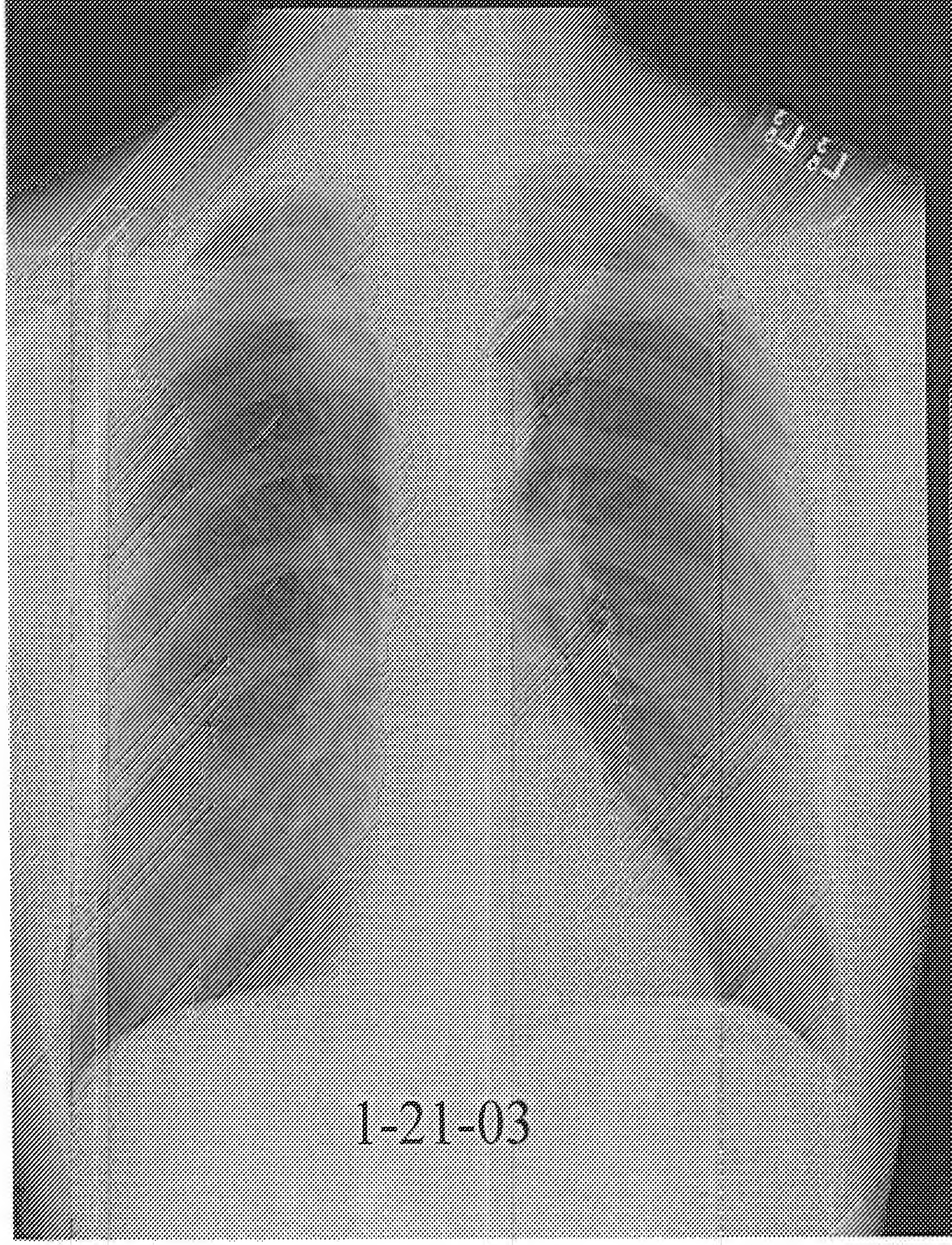
	Pathology	N	Without TS	With TS	ΔA_z	p
Difazio[1]	Mixed	50	.79	.89	+.09	.00004
Uozumi[2]	Nodules	42	.88	.96	+.08	<.05
Johkoh[3]	Nodules	30	.86	.91	+.05 Residents	.003
			.96	.93	-.03 Experts	.36
Kakeda[4]	Nodules	20	.83	.96	+.12 Residents	.027
			.92	.98	+.06 Experts	
Tsubamoto[5]	Pneumonia	60	.92	.95	+.03	<.05

Current CXR



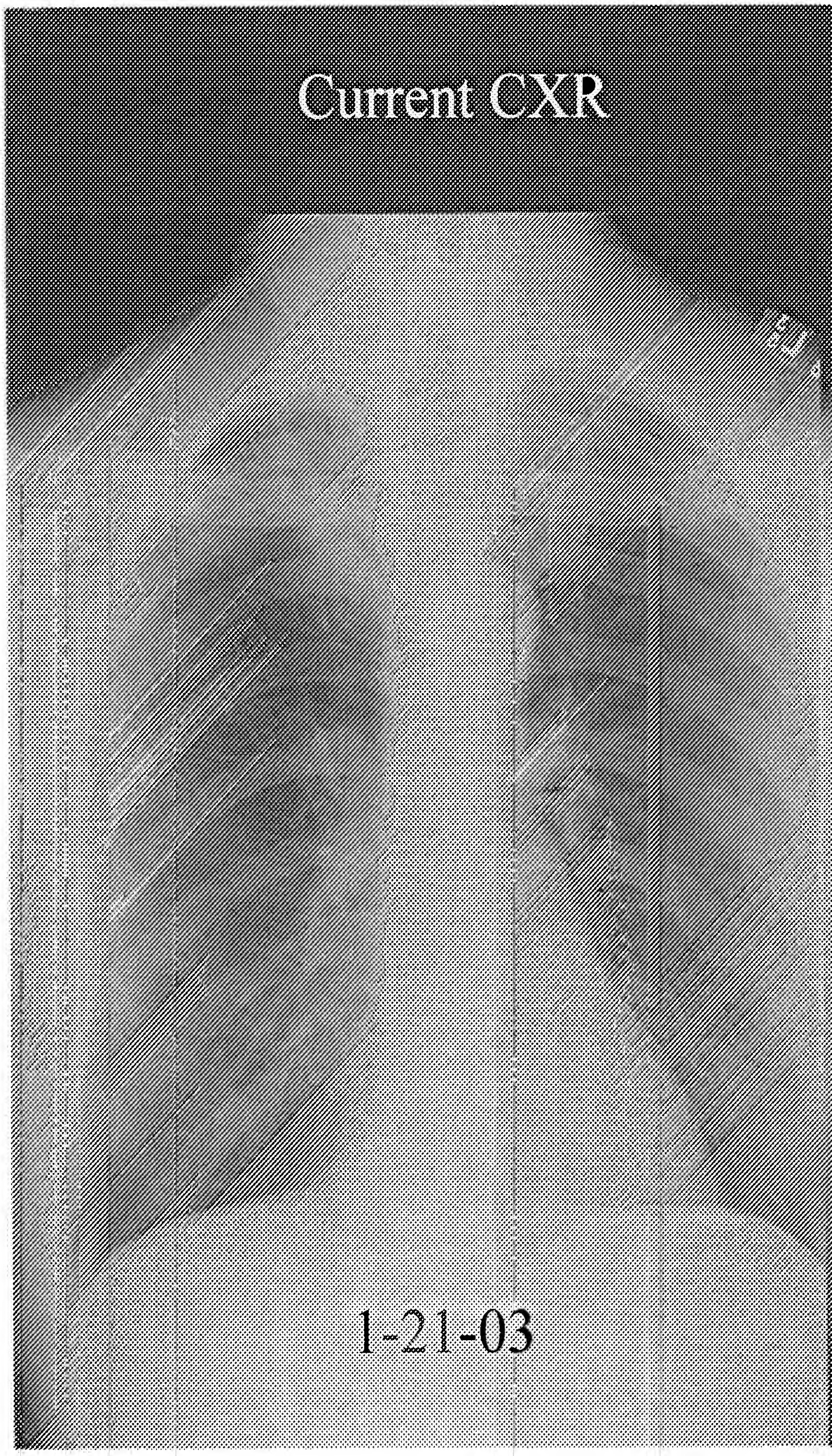
1-21-03

Current CXR



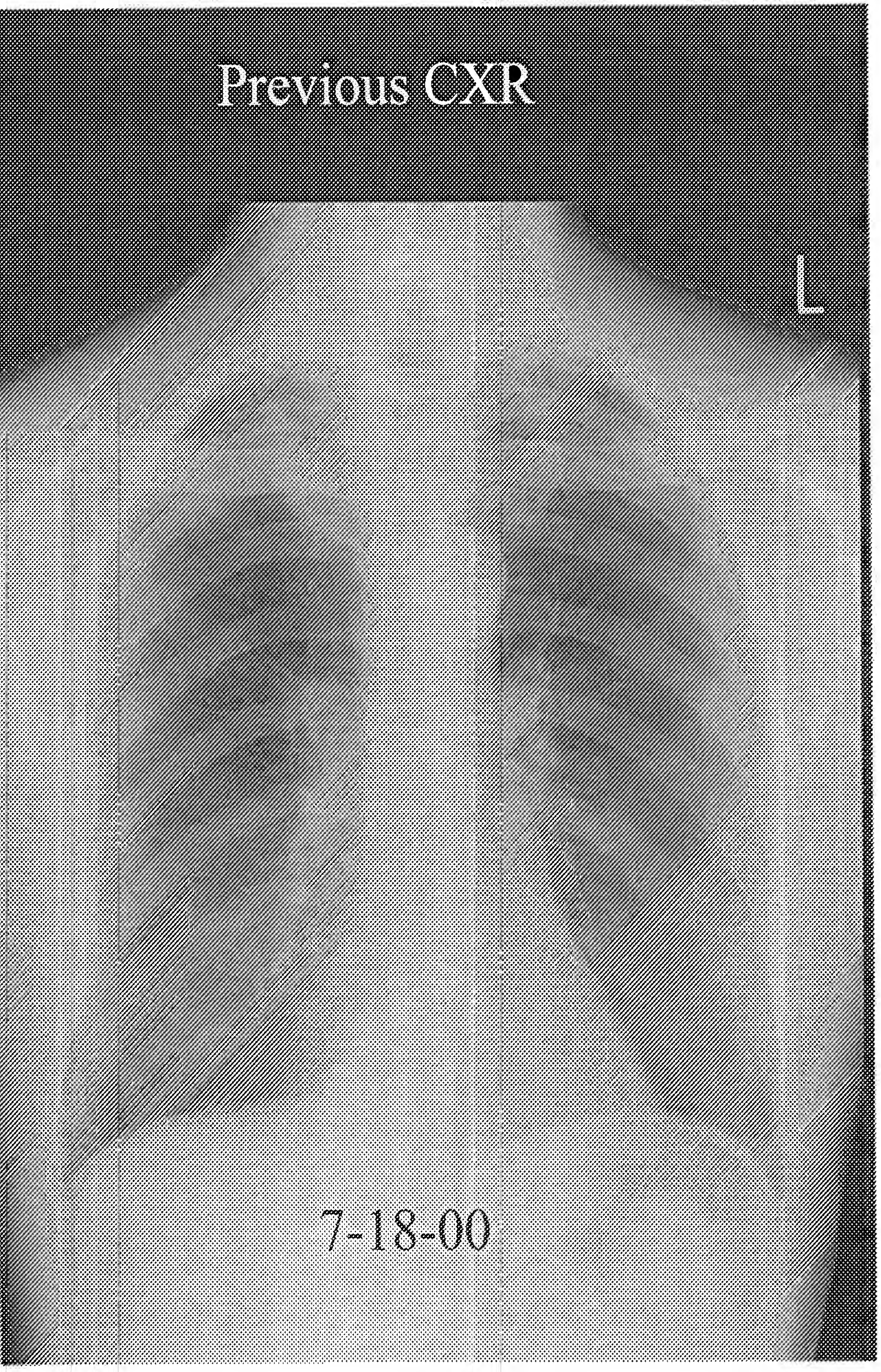
1-21-03

Energy Subtraction
OR
Temporal Subtraction ?



Current CXR

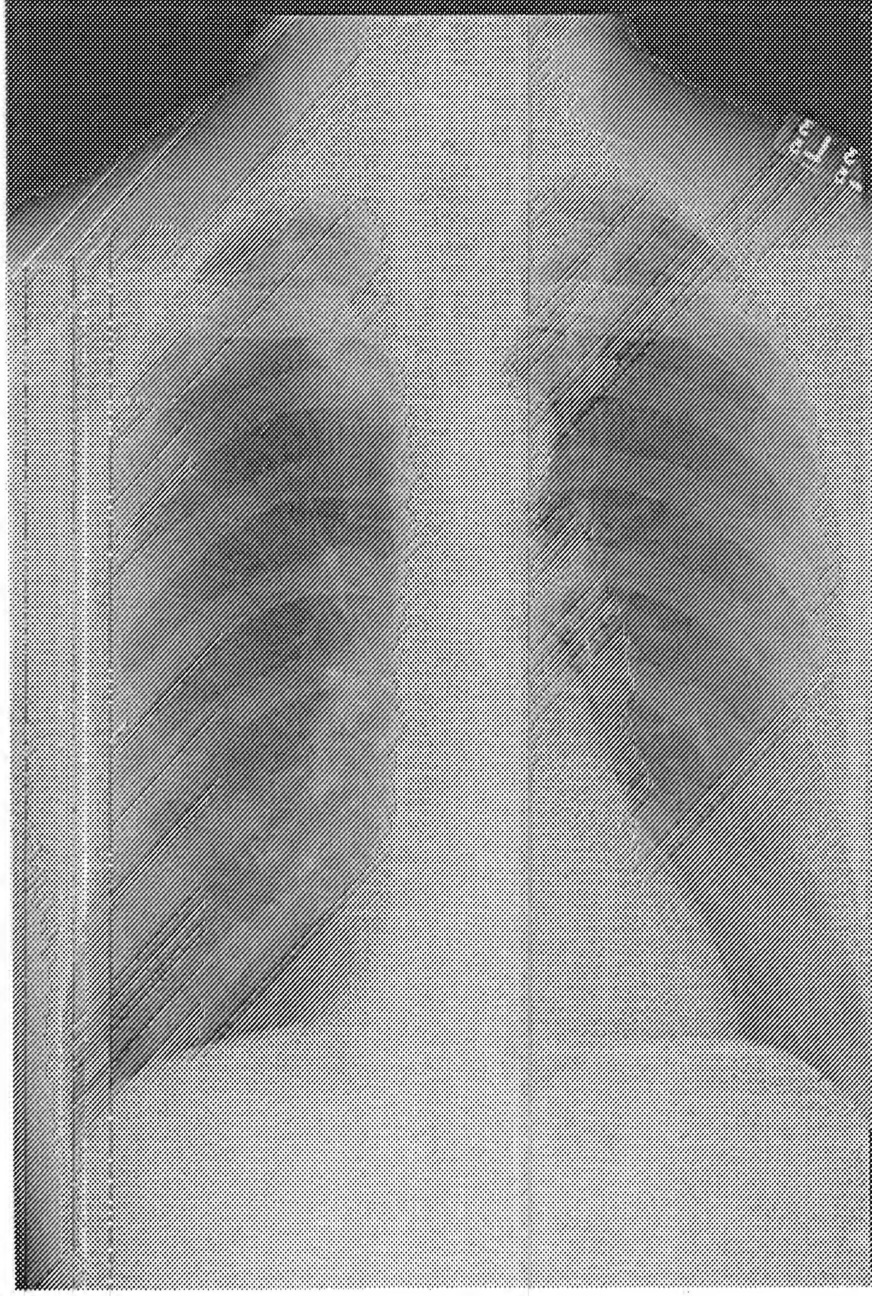
1-21-03



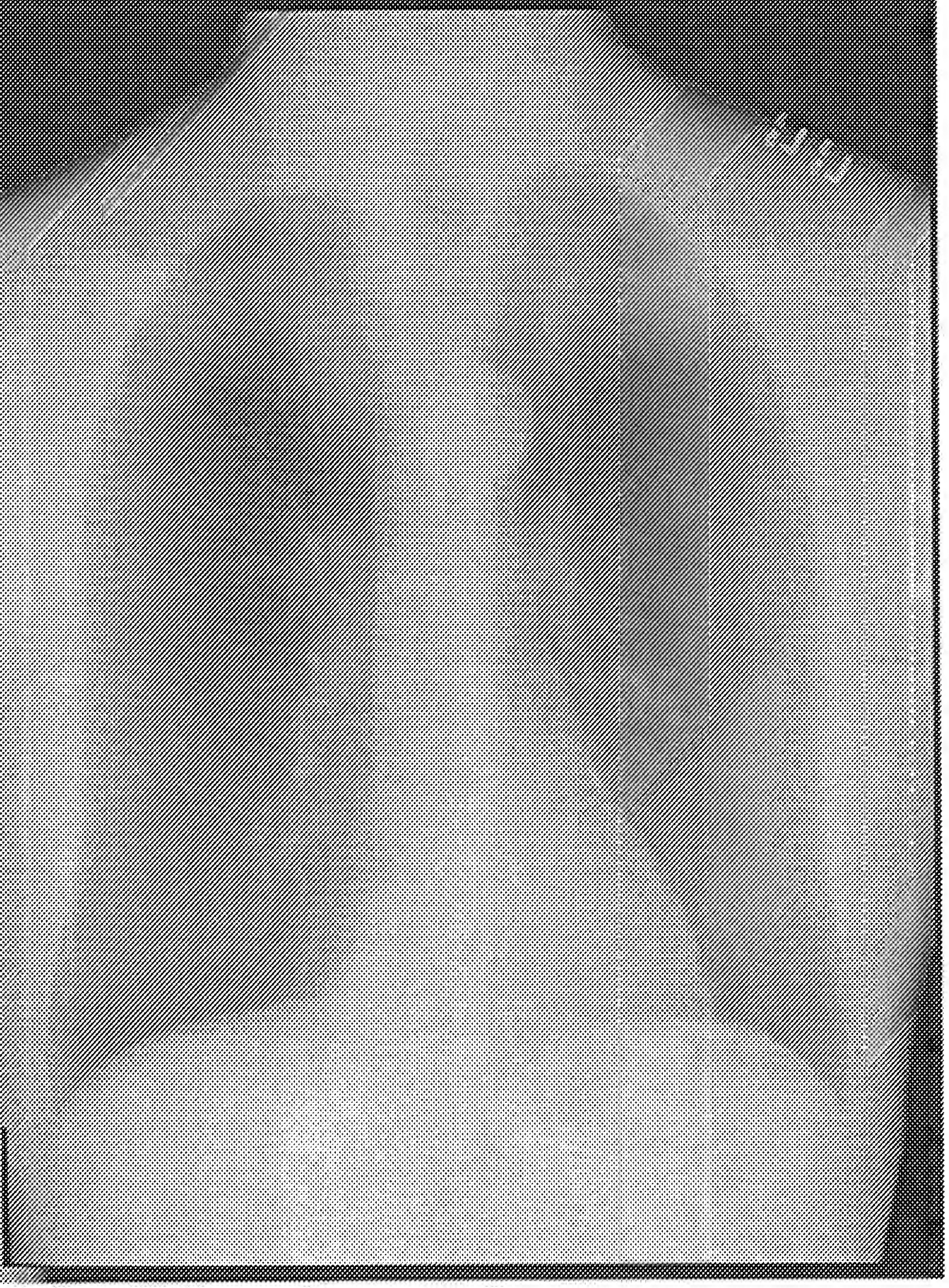
Previous CXR

7-18-00

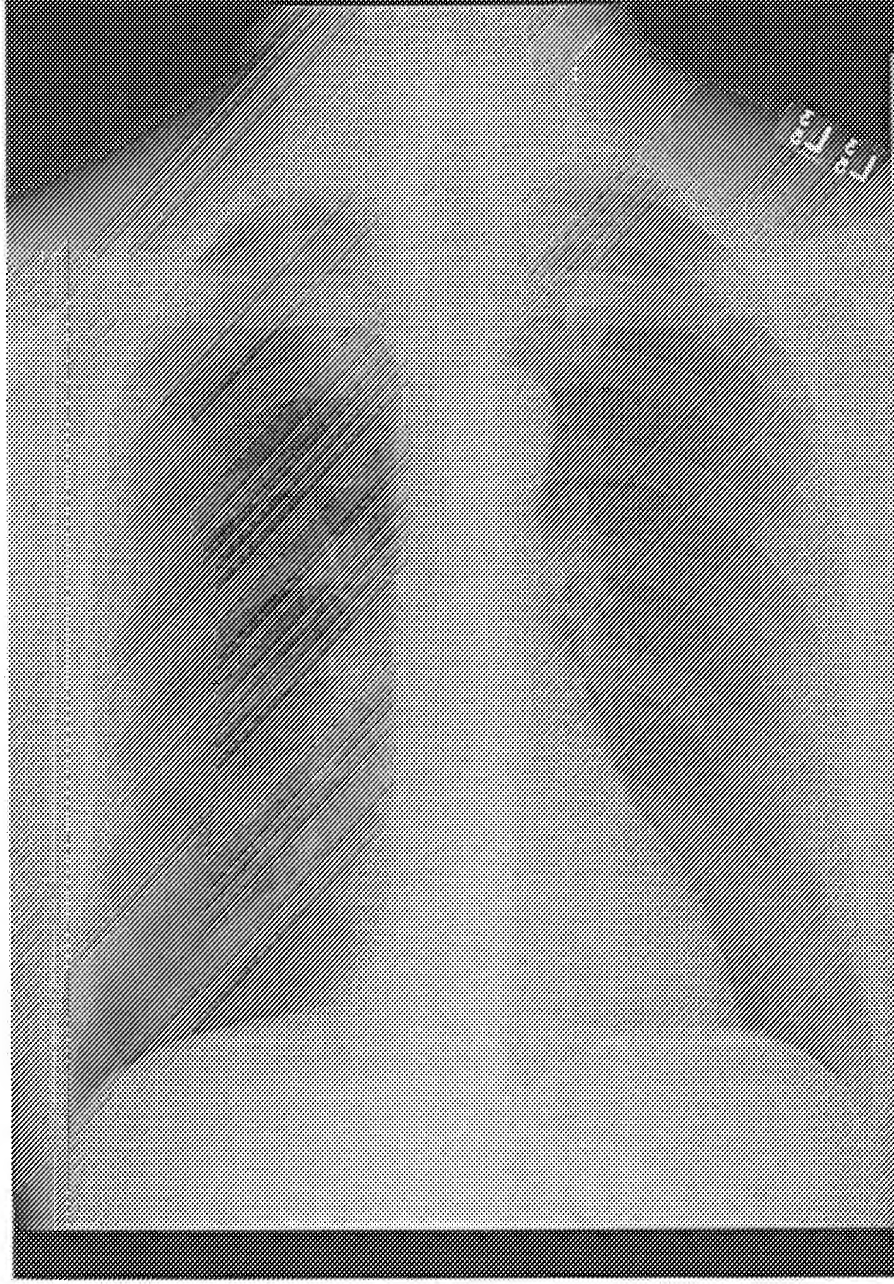
Current CXR



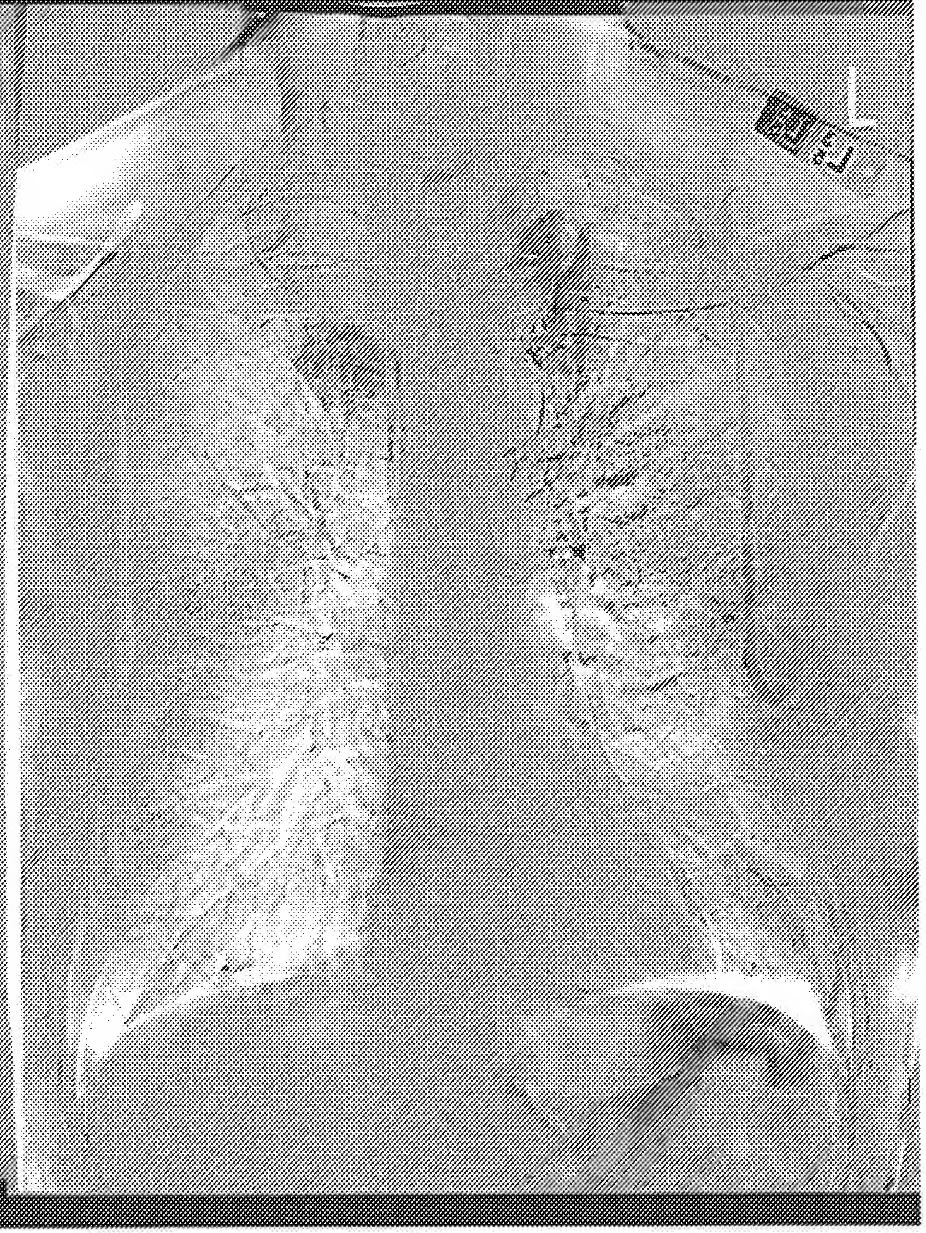
Energy Subtraction CXR



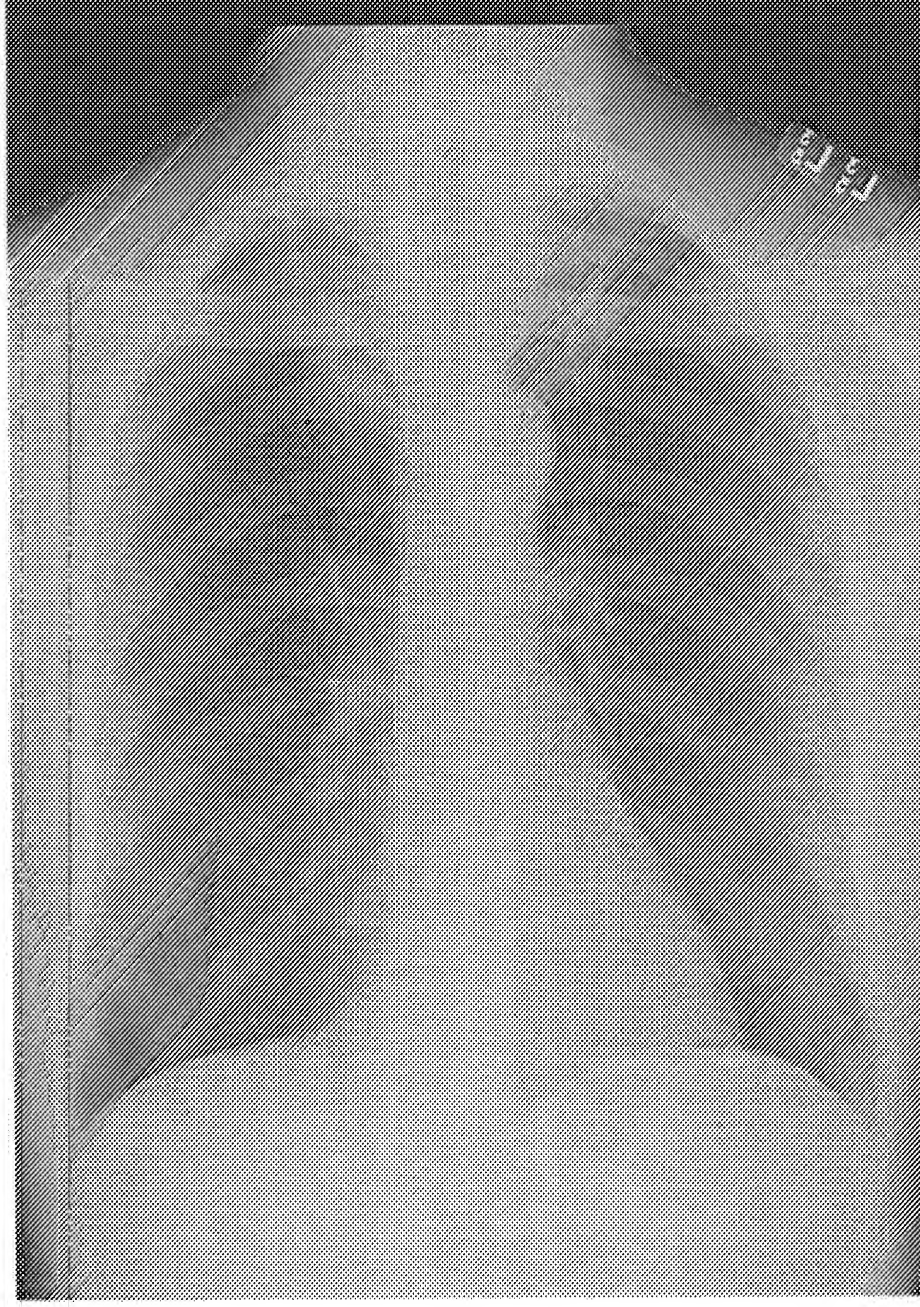
Current CXR



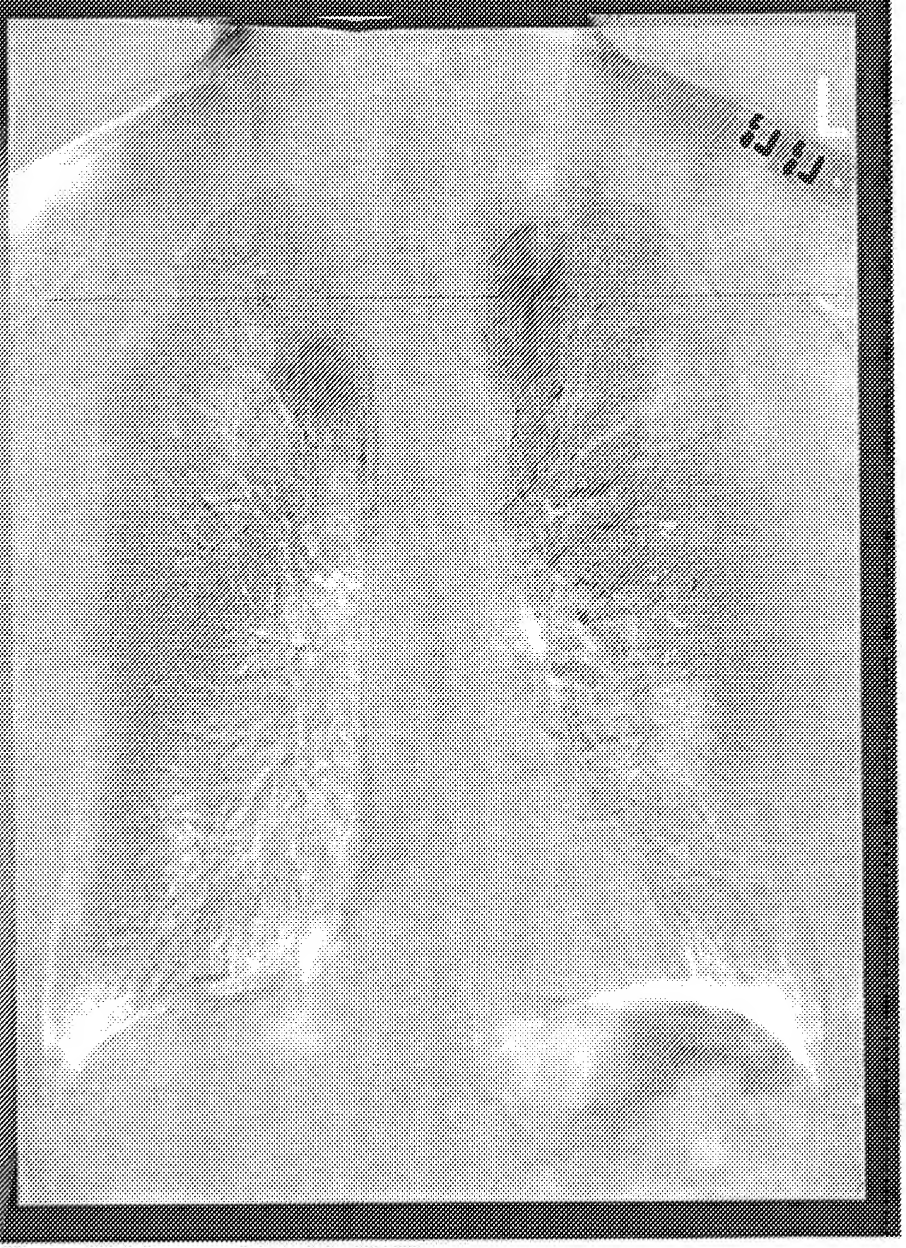
Temporal Subtraction CXR



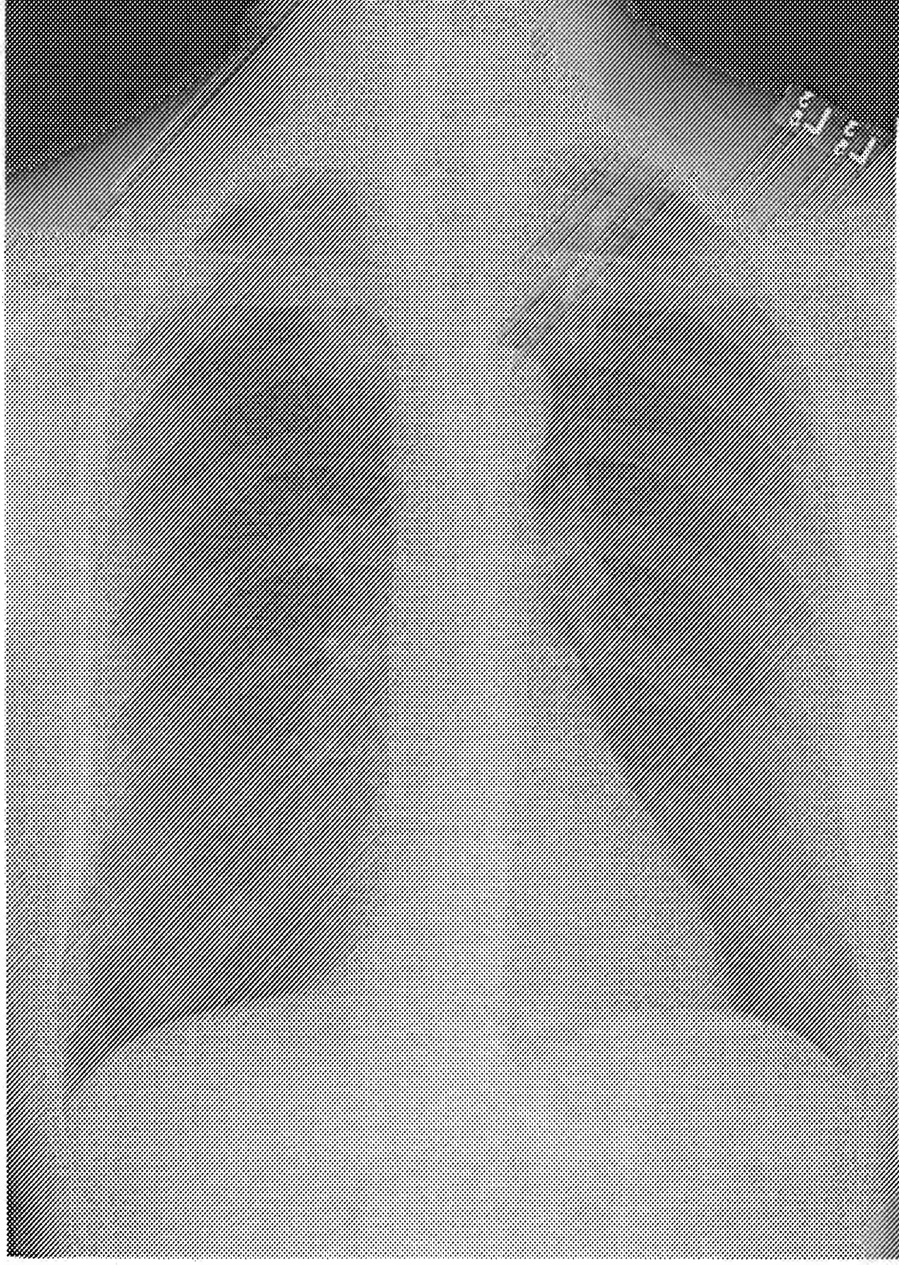
Current CXR



Temporal Soft Tissue Subtraction CXR

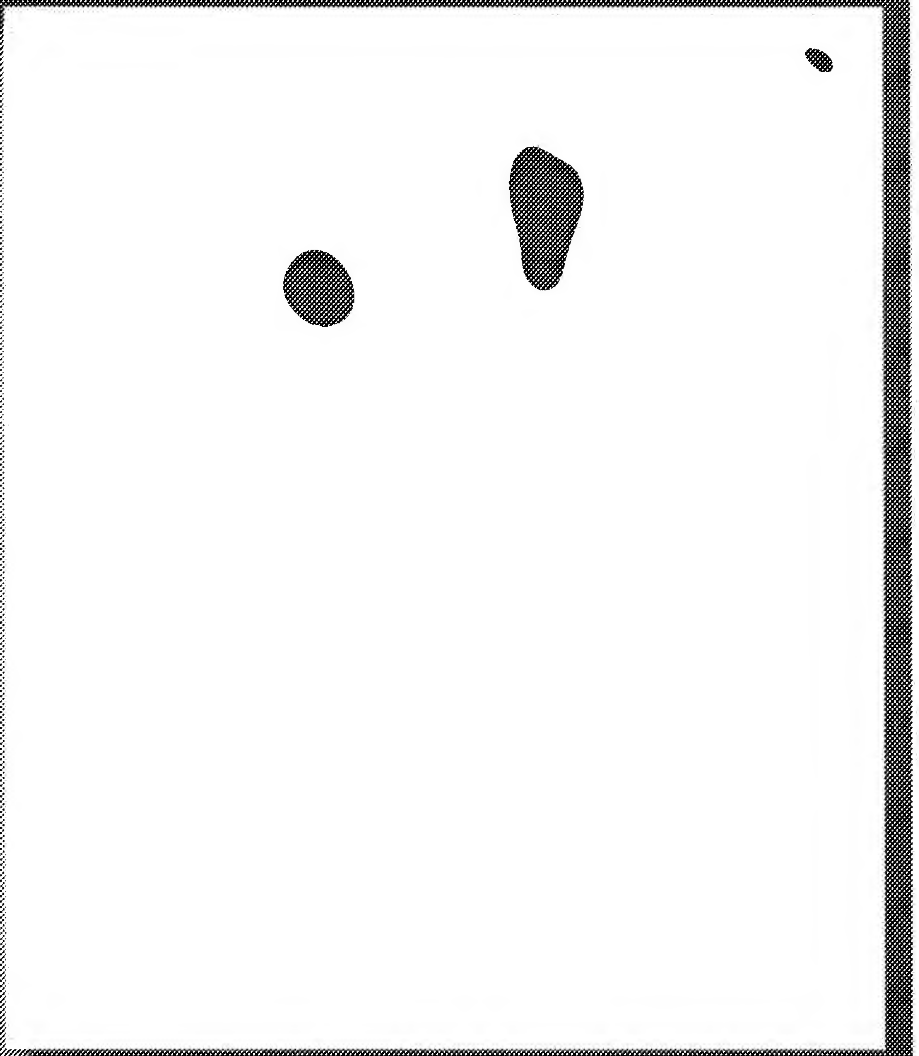


Current CXR



Temporal Soft Tissue Subtraction CXR

Gaussian Blur + Narrow Window



Current CXR

Temporal Soft Tissue Subtraction CXR

Gaussian Blur + Narrow Window